

# CLUBE

# Z

80



**Abril/84**

**N.º 19**

## NESTE NÚMERO

INT. À LINGUAGEM MÁQUINA (Cont.)	1
INT. À LINGUAGEM MÁQUINA (Recapitulação)	4
ENCICLOPÉDIA BASIC (Cont.)	5

### Programas ZX81/Spectrum

Estratégica	5
Sonar	7
Caracteres Gráficos	8
Animação no Spectrum	11
Perseguição	15
Rotinas de Aplicação no Spectrum	16
Caracteres Gráficos	17
Transistores	18
NOVOS PROGRAMAS	20
MERCADO Z80	21

Por limitação de espaço, só no próximo número iniciaremos uma nova rubrica de FERNANDO PRECES: **CONVERSÃO DE PROGRAMAS DO ZX81 → SPECTRUM**

### No Interior:

Cupão de Inscrição



# INTRODUÇÃO À LINGUAGEM MÁQUINA

ZX81/SPECTRUM

Autor: FERNANDO PRECES

(Cont. dos números anteriores)

## PARTE II — SISTEMAS DE NUMERAÇÃO

### 2.1. — Introdução

Hoje, como no passado, o homem na sua infância aprende a contar pelos dedos. É lógico que o seu cérebro utilize sempre, como base de cálculo, o sistema decimal e que oponha resistência ou sinta dificuldade no tratamento de sistemas numéricos noutra base de agrupamento.

No entanto, a mais poderosa máquina por ele criada até hoje (o computador digital), não pode, por razões tecnológicas, pensar, dentro dum sistema de base 10, sem o auxílio dum programa-monitor especializado, que adapte a linguagem do homem e o seu sistema numérico à sua linguagem.

Essa adaptação, porém, é paga por um alto preço: a lentidão de resposta; a ocupação indiscriminada de milhares de bytes de memória; e a limitação de algumas funções da máquina. Para ultrapassar tal situação, deve o utilizador mais experimentado libertar-se, tanto quanto possível, de uma parte importante das instruções de tais linguagens, e tentar estabelecer contacto com a máquina numa linguagem semi-directa, indicada pelo fabricante, no respectivo manual. E é aqui que surge a necessidade de sabermos trabalhar com outros sistemas de numeração.

### 2.2. — Sistema Binário

Os computadores digitais usam um sistema de numeração na base 2, vulgarmente chamado "sistema binário". Utilizam, portanto, uma concepção numérica com apenas 2 algarismos: ("0" e "1"). Os sinais digitais por eles elaborados, são o resultado dos dois estados que os seus circuitos podem discriminar.

Suponha o leitor que vai ensaiar um painel com 4 lâmpadas ligadas em paralelo a uma fonte de tensão adequada, isoladas cada uma por um interruptor, tal como mostra a figura 2.1.

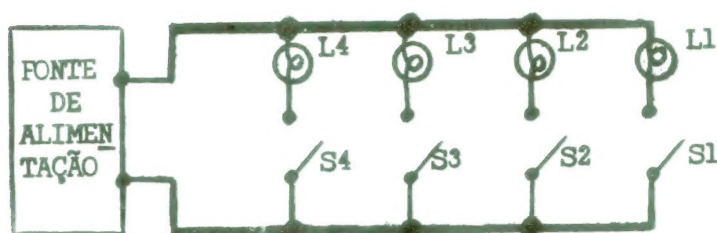


FIGURA 2.1. — Painel com 4 lâmpadas ligadas em paralelo a uma fonte de tensão

A quantidade de informação, resultante da manipulação dos 4 interruptores, é a seguinte:

	Em Binário	Em Decimal
<b>4 lâmpadas apagadas</b>	0000	0
L <sub>1</sub> acesa	0001	1
L <sub>2</sub> acesa	0010	2
L <sub>1</sub> e L <sub>2</sub> acesas	0011	3
L <sub>3</sub> acesa	0100	4
L <sub>3</sub> e L <sub>1</sub> acesas	0101	5
L <sub>3</sub> e L <sub>2</sub> acesas	0110	6
L <sub>3</sub> , L <sub>2</sub> e L <sub>1</sub> acesas	0111	7
L <sub>4</sub> acesa	1000	8
L <sub>4</sub> e L <sub>1</sub> acesas	1001	9
L <sub>4</sub> e L <sub>2</sub> acesas	1010	10
L <sub>4</sub> , L <sub>2</sub> e L <sub>1</sub> acesas	1011	11
L <sub>4</sub> e L <sub>3</sub> acesas	1100	12
L <sub>4</sub> , L <sub>3</sub> e L <sub>1</sub> acesas	1101	13
L <sub>4</sub> , L <sub>3</sub> e L <sub>2</sub> acesas	1110	14
<b>4 lâmpadas acesas</b>	1111	15

Conclui-se deste ensaio que as 4 lâmpadas podem indicar **16** situações diferentes, numeradas à direita em decimal (0 a 15).

Depois deste exemplo, poderá o leitor facilmente concluir que, se ligar nesse painel 8 lâmpadas pelo mesmo processo, vai conseguir encontrar **256** permutações possíveis, que poderá numerar em decimal de (0 a 255). Assim:

com 4 lâmpadas ( 16 combinações)      0000 a 1111  
com 8 lâmpadas (256 combinações) 00000000 a 11111111

No sistema decimal, as posições que os algarismos ocupam na formação dum número, têm o peso de:

$10^0 = 1$  (para a unidade)  
 $10^1 = 10$  ( » » dezena )  
 $10^2 = 100$  ( » » centena )  
 $10^3 = 1000$  ( » o milhar )  
 etc.

No sistema binário, os pesos são:

$2^0 = 1$  (para o bit 1)  
 $2^1 = 2$  ( » » » 2 )  
 $2^2 = 4$  ( » » » 3 )  
 $2^3 = 8$  ( » » » 4 )  
 $2^4 = 16$  ( » » » 5 )  
 etc.

Para converter o binário para decimal, tendo em conta os pesos das posições dos diversos dígitos, basta realizar a operação abaixo exemplificada.



Vamos escolher um número binário ao acaso: 10110

Peso do dígito	2 <sup>4</sup>	2 <sup>3</sup>	2 <sup>2</sup>	2 <sup>1</sup>	2 <sup>0</sup>
Número:	1	0	1	1	0

$$1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 \\ = 16 + 4 + 2 = 22 \text{ (decimal)}$$

Eis um pequeno programa em Basic, tão aproximado quanto possível da operação acabada de realizar, que converte números binários em decimais.

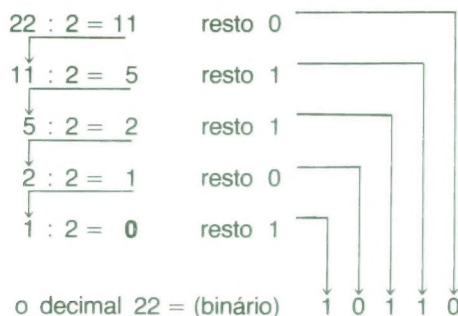
Os possuidores do ZX81, terão apenas de substituir o símbolo ((↑) elevado a) por ((\*) código 216), na linha 40.

```
5 REM SISTEMAS DE NUMERAÇÃO — Programa 1
10 REM CONVERSÃO NÚMERO BINÁRIO EM DECIMAL
15 LET DECIMAL = 0
20 REM INTROD. NUM. BINÁRIO
25 INPUT B$
30 FOR P = LEN B$ TO 1 STEP - 1
40 LET D = (VAL B$ (P)) * (2 ↑ (LEN B$ - P))
50 LET DECIMAL = DECIMAL + D
60 NEXT P
80 PRINT AT 10,4; "DECIMAL = "; DECIMAL
```

Para converter um número decimal num número binário, sem passar por outra base de numeração, o método é um pouco mais moroso. Vejamos o exemplo, aproveitando o número decimal da operação anterior.

(Decimal) 22 = (Binário) 10110

Vamos executar uma divisão sucessiva por 2. O quociente obtido na primeira divisão, volta a ser dividido por 2 e assim por diante, até ser alcançado um quociente 0.



Um pequeno programa em Basic que pode converter números decimais entre 0 e 65535 em números binários, auxiliará o leitor, efectuando as divisões acima referidas.

```
100 REM CONVERSÃO DE UM NÚMERO
DECIMAL PARA BINÁRIO - PROGRAMA 2
110 LET X=0
115 LET B$="0000 0000 0000 0000"
120 REM INTROD. NÚMERO DECIMAL
125 INPUT N
130 LET C=INT (N/2)
140 LET D=N-2*C
150 LET N=C
160 LET B$(16-X)=STR$ D
170 LET X=X+1
175 IF N<>0 THEN GO TO 130
180 PRINT AT 10,4; "BINÁRIO="; B$
```

NOTA: Trabalha nos 2 computadores ZX81 e Spectrum

Temos de reconhecer que, deixar de trabalhar num sistema com 10 algarismos, com o qual estamos totalmente identificados, começar com outro que tem apenas 2 algarismos, necessitando de **16 bits** para representar um número decimal de 5 dígitos, é realmente uma estopada de todo o tamanho. Mas a situação é ultrapassável se introduzirmos de permeio um sistema de numeração cuja base seja múltipla da base 2.

### 2.3. — Sistema numérico hexadecimal

Este sistema utiliza 16 símbolos diferentes; como o sistema decimal é composto por 10, os restantes são representados pelas seis primeiras letras do nosso alfabeto (ver fig. 2.2.).

Decimal	Hexadecimal	Binário
0	00	000000
1	01	000001
2	02	000010
3	03	000011
4	04	000100
5	05	000101
6	06	000110
7	07	000111
8	08	001000
9	09	001001
10	0A	001010
11	0B	001011
12	0C	001100
13	0D	001101
14	0E	001110
15	0F	001111
16	10	010000
17	11	010001
18	12	010010
19	13	010011
20	14	010100
21	15	010101
22	16	010110
23	17	010111
24	18	011000
25	19	011001
26	1A	011010
27	1B	011011
28	1C	011100
29	1D	011101
30	1E	011110
31	1F	011111
32	20	100000

FIGURA 2.2 — As 3 bases de numeração

A base 16 é múltipla à 4.<sup>a</sup> potência, da base 2 ( $2^4 = 16$ ), o que torna muito fácil a conversão de Binário a Hexadecimal. Para exemplo, tomemos o maior número binário de 8 bits, que iremos separar em grupos de 4.

1111 1111  
2.º grupo 1.º grupo

Pelo quadro da figura 2.2, verifica-se que o Binário (1111), corresponde ao Hexa (F). Assim:

2.º grupo 1111 (b) = F (h)  
1.º grupo 1111 (b) = F (h)

NOTA: O (b) quer dizer Binário e o (h) Hexadecimal.



Ao número 1111 1111 (b) corresponde o número FF (h). Tal como fizemos com as outras bases de numeração, vamos agora considerar o **peso** de cada algoritmo do sistema Hexa.

$$16^0 = 1, 16^1 = 16, 16^2 = 256, 16^3 = 4096, \text{ etc.}$$

O número FFFF (h), vai servir para exemplo na conversão a número decimal.

Peso ou posição	(h)	Operação	Resultado
(16 <sup>3</sup> ) 4.º algoritmo	F	15 * 4096	61440
(16 <sup>2</sup> ) 3.º »	F	15 * 256	3840
(16 <sup>1</sup> ) 2.º »	F	15 * 16	240
(16 <sup>0</sup> ) 1.º »	F	15 * 1	(+) 15

**Total** → 65535

Para a passagem de decimal a hexa, vamos usar um processo idêntico ao utilizado pelos computadores para guardar os números.

O número 65535 vai servir como exemplo. A primeira operação consiste em dividi-lo por 256.

$$65535/256 = \underbrace{255}_{\text{Parte inteira}} \bullet \underbrace{9960975 \dots}_{\text{Parte fraccionária}}$$

Ao número encontrado como quociente, extrai-se de imediato a parte inteira (255), a que vamos chamar o byte mais significativo (HIGT BYTE). De seguida multiplica-se a parte fraccionária por 256.

$$\bullet 9960975 \dots * 256 = 255$$

Ao número obtido na 2.ª operação, vamos chamar byte menos significativo (LOW BYTE).

Recorrendo ao quadro da fig. 2.3, que evita mais contas, encontramos facilmente a correspondência entre o decimal (255) e o hexa (FF). Substituindo os valores, temos:

FF (h)	FF (h)
HIGT BYTE	LOW BYTE

Resultando, depois da função:

$$\text{FFFF (h)} = 65535$$

Outro exemplo para praticarmos:

Decimal 32767

$$32767/256 = 127 \bullet 9960975 \dots$$

$$\bullet 9960975 \dots * 256 = 255$$

$$\text{HIGT BYTE} = 127, \text{ LOW BYTE} = 255$$

$$127 \text{ (d)} = 7F \text{ (h)}$$

$$255 \text{ (d)} = FF \text{ (h)}$$

$$32767 \text{ (d)} = 7F FF \text{ (h)}$$

A passagem de hexa a binário, não oferece qualquer problema, visto que a cada algoritmo hexa, corresponde sempre um grupo de 4 bits em binário (ver quadro da figura 2.2).

16 <sup>3</sup> 1000 CHR	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
A	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
B	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
C	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
D	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
E	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
F	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
A	1	0	1	0
B	1	0	1	1
C	1	1	0	0
D	1	1	0	1
E	1	1	1	0
F	1	1	1	1

FIGURA 2.3 — a) Conversão Hexa-decimal — b) Conversão Hexa-binário



# INTRODUÇÃO À LINGUAGEM MÁQUINA

## RECAPITULAÇÃO DE JOGOS

FERNANDO PRECES

### JOGOS DO FINAL DO 1.º CAPÍTULO

(v. n.º 15, Dezembro — págs. 2 e 3  
n.º 16, Janeiro — págs. 2 a 5)

```

1 REM E:RND? TAN = ?*E (RND-
F/ RUN ( NEXT TAN XX
3 GOSUB 1000
10 LET S=0
20 LET A=CODE "<"
30 LET B=A
50 LET Z=S
100 FOR D=3 TO A*A
110 PRINT AT B,Z;
120 IF USR 16514=A THEN GOTO 30
0
130 PRINT "O";
140 IF INKEY$="F" THEN GOSUB 50
0
190 LET B=B-(INKEY$="7" AND B)+
(INKEY$="8" AND B*A)
200 PRINT AT RND*A,17;"<"
210 LET C=USR 16521
220 NEXT D
300 PRINT " ";S;" PONTOS"
310 FOR T=0 TO 300
320 NEXT T
325 CLS
330 PRINT AT 12,0;"QUER CONTINU
AR? (DIGA S OU N).";
340 INPUT L$
345 CLS
350 IF L$="S" THEN GOTO 10
355 PRINT AT 12,0;"ATE A PROXIM
A E... OBRIGADA.";U
500 FOR C=Z TO PI
510 IF USR 16514=A THEN LET S=S
+1
520 PRINT "-";
530 NEXT C
540 PRINT AT B,Z;"O "
550 RETURN
1000 PRINT AT 1,5;"BATALHA NO ES
PAÇO"
1010 PRINT "A SUA NAVE E ATAC
ADA POR CENTE-"
1020 PRINT "NAS DE PEQUENAS N
AVES, CUJO FO-"
1030 PRINT "GO E INOFENSIVO M
AS, PERIGOSAS"
1040 PRINT "AO CHOQUE, A TECL
A "7" DESVIA A"
1050 PRINT "NAVE PARA CIMA, A
"8" PARA BAIXO"
1060 PRINT "E A TECLA "F" DI
SPARA."
1070 PRINT "BOA SORTE.;"
1080 PRINT "PRIMA "N/L" PARA
COMEÇAR."
1090 INPUT L$
1095 CLS
1099 RETURN
2000 SAVE "JOGO 2"
2010 RUN

```

```

5 REM JOGO 2
10 REM PROGRAMA EM BASIC
20 SLOW
30 CLS
40 GOSUB 100
50 FOR A=1 TO 18
60 PRINT AT A,0;" ";AT A,31;"

```

```

70 NEXT A
80 GOSUB 100
90 GOTO 140
100 FOR A=0 TO 31
110 PRINT " ";
120 NEXT A
130 RETURN
140 LET ED=1
150 LET CB=1
160 LET X=3
170 LET Y=INT (RND*33+8)
180 PLOT X,Y
190 IF INKEY$="R" THEN RUN
200 IF X=2 OR X=61 THEN LET ED=
-ED
210 IF Y=6 OR Y=41 THEN LET CB=
-CB
220 UNPLOT X,Y
230 LET X=X+ED
240 LET Y=Y+CB
250 GOTO 180
500 SAVE "JOGO 2"
510 RUN

```

```

10 REM Y$=4NOT ( CLEAR TAN Y<
?LN PRINT Y$NOT TAN LN F?LN E<L
N <RND>VAL : LN <RNDAT VAL :3LN
<RNDAT ( LIST LN <RNDTAN LN <RND
DTAN

```

```

11 REM COPY = BY MWRNDM<RNDJWM
""AND GOSUB ?""RNDY<MKANDLN <LN
< FOR 5 CLS LOAD < GOSUB 7003
5< CLEAR < GOSUB 7003 U""AND RET
URN <C< RETURN X4<U<RNDJWM<RNDUR
T AND RETURN <C< RETURN D4<U<RND
JWM<RND GOSUB ?""RNDY<MKANDLN <
5<RND""AND<F=<45678901345
67890

```

```

20 LET K=USR 16533
30 POKE 16577,INT (RND*33+8)
35 PRINT AT 10,8;" "
36 PLOT 20,20
37 PLOT 25,10
40 LET K=USR 16578
100 REM JOGO 2 EM CODIGO MAQUIN
A
500 SAVE "JOGO 2"
510 RUN

```

### AOS LEITORES:

FERNANDO PRECES enviou-nos também uma listagem com o conjunto dos programas de todo o 1.º capítulo da rubrica INTRODUÇÃO À LINGUAGEM MÁQUINA, ZX81. Publicá-la-emos se os leitores nos manifestarem o seu interesse — ficamos a aguardar a v/ opinião.

### TROCA DE PROGRAMAS

ANTÓNIO NUNES enviou ao CLUBE Z80 uma lista dos 76 programas que possui para trocar c/ sócios por outros que não tem. Não publicamos essa lista por ser muito extensa e os interessados podem pedi-la ao CLUBE, enviando 20\$00 em selos CTT's, ou contactar c/ o sócio:

ANTÓNIO NUNES, Rua do Til, 72 — 9000 FUNCHAL.



## ENCICLOPÉDIA BASIC — ABRIL/84

(Cont. dos números anteriores)

### AUTO — comando

Este comando fornece um meio expedito de inserir números de linhas no programa já escrito. Quer a linha inicial, quer o intervalo entre números de linha, podem ser automáticos (por exemplo Hewlett Packard).

#### EXEMPLO:

AUTO 100,10 coloca o número inicial em 100 e o intervalo entre linhas, será de 10 em 10.

ESTE COMANDO É USADO NO INÍCIO OU NO DECORRER DA INTRODUÇÃO DO CÓDIGO BASIC, MAS NÃO DENTRO DE UM PROGRAMA.

### BASE — instrução

Trata-se de uma instrução pertencente à norma ANSI e que é possível encontrar no Control Data ou no New Brain (option base).

A sua utilização destina-se a definir o valor mais baixo de um ARRAY (quadro ou tabela ou matriz), como 0 ou 1.

#### EXEMPLO:

```
10 BASE 0
20 DIM A(15)
```

A instrução BASE define o array desde A(0) até A(15), ou seja 16 elementos.

Muitos computadores estabelecem os arrays entre 0 e 10, ou seja 11 elementos, sem que exista essa declaração inicial.

A instrução BASE só pode ser usada antes do DIMensionamento e antes da manipulação das variáveis indexadas.

Alguns dialectos de BASIC permitem que a instrução BASE seja limite inferior das variáveis indexadas.

#### EXEMPLO:

```
10 BASE 6
20 DIM A(20)
```

definiria o array como 15 elementos, desde A(6) até A(20).

### BREAK — comando

Trata-se de um comando que permite interromper o curso de um programa. Algumas máquinas permitem usar este comando no interior do programa, de modo a interromper o programa antes das linhas especificadas.

#### EXEMPLO:

10 BREAK 40,80,120 significa que o programa se detém antes das linhas 40, 80 e 120, de modo a que o operador possa verificar ou alterar determinadas condições. (De um modo similar ao STOP).

Regra geral, após uma interrupção originada por BREAK, poderá recomeçar, usando o comando CONTinue ou CONTinue. Os valores das variáveis não são afectados por este comando. Em muitas máquinas, o comando BREAK está implementado numa das teclas.

### BYE — comando ou instrução

Este comando é usado frequentemente para sair do BASIC. É um caso típico dos sistemas em TIME-SHARING, que aceitam o BYE como sinal do fim da execução.

NO CASO DAS MÁQUINAS ATARI — SOL por exemplo, o comando BYE serve para chamar o Sistema Operativo (disk operating system).

Em alternativa alguns sistemas permitem usar GOODBYE para significar fim da tarefa e SYSTEM para chamar o DOS.

(Cont. no próximo número)

## ESTRATÉGIA

ZX81 16 K

Autor: FERNANDO PRECES  
Sacavém

```
0 REM PROGRAMA MODIFICADO POR
1 ALMEIDA PRECES EM 20/7/1982
2 REM "S"
3 GOSUB 5000
4 INPUT T
5 CLS
6 IF T=1 THEN GOTO 8
7 IF T=2 THEN GOTO 2000
8 LET C=168
9 LET H=173
10 DIM A(25)
11 DIM B(7)
12 FOR A=3 TO 5
13 LET A(A)=C
14 NEXT A
15 FOR A=21 TO 23
16 LET A(A)=H
17 NEXT A
```

```
110 LET B(1)=5
120 LET B(2)=4
130 LET B(3)=6
140 LET B(4)=-1
150 LET B(5)=-3
160 LET B(6)=-8
170 GOSUB 1000
180 FOR A=20 TO 1 STEP -1
190 FOR J=1 TO 5
200 LET B=INT (RND*3)+1
210 IF 5*INT (A/5)=A AND B=3 THEN
220 GOTO 210
230 IF A(A)=0 AND A(A+B(B))=0 THEN
240 GOTO 270
250 NEXT J
260 NEXT A
270 FOR A=7 TO 20
280 FOR B=4 TO 5
```



```

235 IF 5*INT (A/5)=A AND B=4 TH
EN GOTO 250
240 IF A(A)=0 AND A(A+B(B))=0 T
HEN GOTO 270
250 NEXT B
260 NEXT A
265 PRINT AT 20,0;"      EU PASSO

```

```

270 LET A(A+B(B))=0
280 LET A(A)=0
290 GOSUB 1000
310 INPUT A$
320 LET D=5*(CODE A$-38)+CODE A
#(2)-28
330 LET E=5*(CODE A$(3)-38)+COD
E A$(4)-28
340 LET A(E)=H
350 LET A(D)=0
360 GOTO 165
1000 LET X=0
1010 LET Y=0
1012 PRINT AT 20,0;"

```

```

1013 LET Z=RND*+RND
1015 PRINT AT 20,0;"-----
-----"
1020 PRINT AT 5,0;"1 2 3 4 5"
1025 PRINT
1030 FOR A=1 TO 25
1040 IF A(A)=0 THEN PRINT " ";
1045 IF A(A)<>0 THEN PRINT CHR$
(A(A));" ";
1050 IF 5*INT (A/5)=A THEN PRINT
" ";CHR$ (A/5+37);
1060 IF A(A)=0 AND A>20 THEN LET
X=X+1
1070 IF A(A)=H AND A<5 THEN LET
Y=Y+1
1080 NEXT A
1090 PRINT "1 2 3 4 5"
1095 PRINT
1100 PRINT "HOMEM ";Y;" COMPUTAD
OR ";X
1101 IF Y>X THEN PRINT "VOCE EST
A GANHANDO"
1102 IF X>Y THEN PRINT "--EU EST
OU GANHANDO"
1105 IF X=4 OR Y=4 THEN GOTO 112
0
1110 RETURN
1120 IF X>Y THEN PRINT "COMPUTAD
OR ";
1130 IF Y>X THEN PRINT "HOMEM";
1140 PRINT "VITORIA COM ";ABS (X
-Y);" PONTOS"
1150 STOP

```

```

2005 LET C=141
2007 LET H=140
2010 DIM A(25)
2020 DIM B(7)
2030 FOR A=3 TO 5
2040 LET A(A)=0
2050 NEXT A
2070 LET A(11)=H
2080 LET A(16)=H
2090 LET A(21)=H
2110 LET B(1)=5
2120 LET B(2)=4
2130 LET B(3)=6
2140 LET B(4)=-4
2150 LET B(5)=-5
2160 LET B(6)=-6
2165 GOSUB 3000
2170 FOR A=20 TO 1 STEP -1
2175 FOR J=1 TO 5
2180 LET B=INT (RND*3)+1
2185 IF 5*INT (A/5)=A AND B=3 TH
EN GOTO 2210
2190 IF A(A)=0 AND A(A+B(B))=0 T
HEN GOTO 2270
2200 NEXT J

```

```

2210 NEXT A
2220 FOR A=7 TO 20
2230 FOR B=4 TO 6
2232 IF (A=6 OR A=11 OR A=16) AN
D B=4 THEN GOTO 2250
2235 IF 5*INT (A/5)=A AND B=4 TH
EN GOTO 2250
2240 IF A(A)=0 AND A(A+B(B))=0 T
HEN GOTO 2270
2250 NEXT B
2260 NEXT A
2265 PRINT AT 20,0;"      EU PASSO

```

```

2270 LET A(A+B(B))=0
2280 LET A(A)=0
2290 GOSUB 3000
2310 INPUT A$
2320 LET D=5*(CODE A$-38)+CODE A
#(2)-28
2330 LET E=5*(CODE A$(3)-38)+COD
E A$(4)-28
2340 LET A(E)=H
2350 LET A(D)=0
2360 GOTO 2165
3000 LET X=0
3010 LET Y=0
3012 PRINT AT 20,0;"

```

```

3013 LET Z=RND*+RND
3015 PRINT AT 20,0;"-----
-----"
3020 PRINT AT 5,0;"1 2 3 4 5"
3025 PRINT
3030 FOR A=1 TO 25
3040 IF A(A)=0 THEN PRINT " ";
3045 IF A(A)<>0 THEN PRINT CHR$
(A(A));" ";
3050 IF 5*INT (A/5)=A THEN PRINT
" ";CHR$ (A/5+37);
3060 IF A(A)=0 AND A>20 THEN LET
X=X+1
3070 IF A(A)=H AND 5*INT (A/5)=A
THEN LET Y=Y+1
3080 NEXT A
3090 PRINT "1 2 3 4 5"
3095 PRINT
3100 PRINT "HOMEM ";Y;" COMPUTAD
OR ";X
3101 IF Y>X THEN PRINT "VOCE EST
A GANHANDO"
3102 IF X>Y THEN PRINT "--EU EST
OU GANHANDO"
3105 IF X=4 OR Y=4 THEN GOTO 312
0
3110 RETURN
3120 GOTO 1120

```

```

5000 PRINT AT 6,9;"GRANDES JOGOS
"
5010 FOR U=0 TO 150
5020 NEXT U
5060 CLS
5070 PRINT AT 1,8;"CERCO ESTRATE
GICO"
5075 PRINT "CADA JOGADOR TEM
3 PEDRAS, O ZX"
5080 PRINT "JOGA COM AS 6 E O HO
MEM COM AS 4"
5085 PRINT "A JOGADA INICIAL PER
TENCE AO ZX"
5090 PRINT "E A FINALIDADE DO JO
GO CONSISTE"
5095 PRINT "EM CERCAR O ADVERSAR
IO IMPEDINDO"
6000 PRINT "ESTE DE ALCANCAR O S
ORDO OPC3TO"
6005 PRINT "DO TABULEIRO, RONDE
VAI PONTUAR,"
6010 PRINT "AS PEDRAS MOUEM-SE E
M TODAS AS"
6015 PRINT "DIRECOES, A DIREITO
E EM DIAGO-"

```



```

6020 PRINT "NAL, EXCEPTO AS QUE
JA CHEGARAM"
6025 PRINT "A PONTUAR. O JOGO TE
M 2 VERSOES"
6030 PRINT "A *1* E A *2*, DEVE
INTRODUZIR O"
6035 PRINT "NUMERO DESEJADO."
6040 PRINT ",,"PRIMA **N/L** PARA

```

```

COMECAR"
6045 INPUT L$
6050 CLS
6060 PRINT AT 7,15;"VERSAO 1"
6065 PRINT AT 10,15;"VERSAO 2"
6070 PRINT ",,"INTRODUZA O NUME
RO"
6099 RETURN

```

## SONAR

ZX81 16 K

Autor: FERNANDO PRECES

Sacavém

```

1 REM "6"
3 DIM A(10,10)
10 DIM B(10,10)
13 GOSUB 1500
16 GOSUB 400
20 GOSUB 500
23 GOSUB 600
26 GOSUB 700
30 GOSUB 1200
33 GOSUB 800
40 GOTO 20
115 STOP
400 LET U=30
401 LET Z=5000
404 FOR E=1 TO 6
405 LET Y=INT (RND*10+1)
410 LET X=INT (RND*10+1)
415 LET A(X,Y)=1
418 LET B(X,Y)=1
420 NEXT E
425 LET C=5
430 LET D=5
440 LET B(C,D)=2
445 LET N=8
499 RETURN
500 PRINT AT 0,0;
505 PRINT "1234567890"
510 FOR Q=1 TO 10
515 PRINT AT Q,13;",";AT Q,13;"
520 IF Q<10 THEN PRINT Q;" ";
522 IF Q=10 THEN PRINT "0 ";
524 FOR P=1 TO 10
525 IF A(Q,P)=1 AND B(Q,P)=2 TH
IN LET N=N-1
526 IF A(Q,P)=1 AND B(Q,P)<>1 T
HEN LET B(Q,P)=4
535 IF B(Q,P)=0 THEN PRINT " ";
540 IF B(Q,P)=1 THEN PRINT "■";
545 IF B(Q,P)=2 THEN PRINT "■";
550 IF B(Q,P)=4 THEN PRINT "+";
560 NEXT P
565 NEXT Q
567 LET U=U-1
568 LET Z=Z-100
569 IF U=0 THEN GOTO 1000
570 PRINT AT 12,4;"SONAR"
575 PRINT AT 0,20;"■"
580 PRINT AT 1,15;"-----"
585 PRINT AT 1,20;"■■■■■■■■■■"
590 PRINT AT 5,21;"■■■■■■■■■■" (*);N
;"")
598 IF N=0 THEN GOTO 1000
599 RETURN
600 FOR T=2 TO 11
630 FOR U=2 TO 11
635 PRINT AT T,U;"■"
660 NEXT U
665 NEXT T
699 RETURN
700 FOR R=1 TO N
735 LET H=INT (RND*10+1)
740 LET I=INT (RND*10+1)
745 IF B(H,I)=2 THEN GOTO 1100

```

```

750 NEXT R
795 RETURN
800 PRINT AT 18,0;"DIRECCAO: (N
OU S) ?"
802 INPUT L$
804 PRINT AT 18,0;"DIRECCAO: (E
OU O) ?"
806 INPUT M$
808 PRINT AT 18,0;"
810 LET B(C,D)=0
825 IF L$="N" THEN LET C=C-1
830 IF L$="S" THEN LET C=C+1
835 IF M$="E" THEN LET D=D+1
840 IF M$="O" THEN LET D=D-1
845 LET B(C,D)=2
850 RETURN
1000 PRINT AT 16,8;"PONTUACAO ";
Z
1010 STOP
1100 FOR O=1 TO 30
1110 PRINT AT 1,18;"TORPEDIADO"
1115 PRINT AT 1,18;" "
1120 PRINT AT 1,18;"TORPEDIADO"
1130 NEXT O
1135 STOP
1200 IF B(X,Y)=4 THEN RETURN
1205 LET A(X,Y)=0
1210 LET B(X,Y)=0
1215 LET T=INT (RND*10+1)
1220 IF T=1 OR T=3 OR T=5 THEN L
ET X=X+1
1225 IF T=2 OR T=4 THEN LET X=X-
1
1230 IF T=6 OR T=8 OR T=10 THEN
LET Y=Y-1
1235 IF T=7 OR T=9 THEN LET Y=Y+
1
1240 IF X>10 THEN LET X=10
1245 IF Y>10 THEN LET Y=10
1246 IF X<1 THEN LET X=1
1247 IF Y<1 THEN LET Y=1
1250 LET A(X,Y)=1
1260 LET B(X,Y)=1
1270 RETURN
1500 PRINT "O SONAR E OS SUBMARI
NOS."
1510 PRINT ",,"VOCE COMANDA UM NA
VIO DE GUERRA,"
1520 PRINT ",,"TEM 6 SUBMARINOS N
O SEU RAIO DE"
1530 PRINT ",,"ACCAO. VOCE DESTRO
I UM SUBMARINO"
1540 PRINT ",,"QUANDO OCUPAR A SU
A CASA, MAS A"
1550 PRINT ",,"TODO O INSTANTE TA
MBEM PODE SER"
1560 PRINT ",,"TORPEDIADO. AO FIM
RECEBE A PON-"
1570 PRINT ",,"TUACAO. BOA SORTE.
"
1580 PRINT ",,"PRIMA **N/L** PARA
COMECAR,"
1590 INPUT L$
1592 CLS
1595 RETURN

```



## CARACTERES GRÁFICOS

ZX81

In. ZX COMPUTING, Dezembro/Janeiro 1984

Trad. e Adapt.: J. MAGALHÃES

O objectivo deste programa é tornar possível a criação de qualquer caracter de tamanho superior ao normal, através do seu armazenamento num «array». No final do programa, o «display» do écran é gravado na posição mais alta da RAM («high memory»), o que vai tornar possível a sua adaptação num outro programa.

A utilização do programa torna-se bastante interessante quer pelos seus efeitos quer pela capacidade de apresentação normal ou em inverse-video.

O programa funciona igualmente com um número de caracteres inferior a 54 e torna possível a adaptação de novos caracteres com a opção de redefinir qualquer caracter.

### COMO FUNCIONA O PROGRAMA

As linhas 20 a 240 referem-se à criação de um caracter gráfico e armazenamento de dados num «array». O «array» C é calculado pela subrotina na linha 640, havendo depois uma verificação da sua existência na linha 100. Quando a definição está completa, faça NEWLINE para introduzir a rotina PLOT que funciona como vamos descrever:

As linhas 260 e 270 indicam a posição inicial, a partir da qual é posicionado o primeiro caracter. A linha 290 chama um caracter. A subrotina na linha 640 converte C\$ numa posição C no «array». Os «bytes» individuais armazenados nesta posição da «string» são então **carregados** («Poked») na primeira das 15 localizações da linha 1 (REM). Esta operação é levada a cabo pelas linhas 330 a 390.

A linha 410 copia o «display» do écran para o ponto mais alto da RAMTOP. A linha 420 coloca o caracter escolhido na posição inicial. As linhas 430 a 470 **estudam** o teclado para uma entrada («Input») com incrementação dos movimentos X-Y ou diminuição dos valores no endereço 16542 e 16544.

A linha 500 copia o écran no regresso da RAMTOP, e dá-se o salto para a linha 420 onde o caracter escolhido é re-apresentado. A linha 470 permite a saída deste «loop», facilitando a apresentação no écran do caracter escolhido, quando há nova execução da linha 410.

Quando a posição do caracter no écran está finalizada, fazendo NEWLINE, entra a rotina de inversão (INVERSE-VÍDEO) pela linha 560. A linha 570 armazena o «écran» na posição mais alta da RAMTOP. A opção final é para sair do programa pela tecla «S», ou NEWLINE para re-entrar na rotina PLOT.

### INTRODUÇÃO DO PROGRAMA

Reserve parte da memória para as rotinas em Código Máquina, dando entrada directa dos seguintes comandos:

```
POKE 16388,0
POKE 16389,125
NEW
```

Isto coloca a RAMTOP em 32 000.

Dê entrada do programa «carregador» Hex. da fig. 1, assegurando-se que a linha 1 contém pelo menos 109 caracte-

res. Quando terminar esta fase, dê entrada do código Hex., como é apresentado na coluna da esquerda da fig. 2, terminando cada linha de código com NEWLINE.

Se ocorrer qualquer erro durante a introdução de dados, fazendo NEWLINE, seleccione «EDIT» (letra «E» em inverse video). A partir deste momento pode fazer a correcção pela entrada correcta do código e continuar a introdução de dados.

Completa esta fase, dê entrada das letras «Z,Z». Apague todas as linhas excepto a linha 1; faça CLEAR. Dê entrada directa do comando DIM A\$(54, 5, 3) para criar um array de 54 caracteres.

Note que esta entrada não necessita de número de linha. Pode agora adicionar as linhas em BASIC apresentadas na fig. 3 (a partir da linha 20).

Depois de introduzida esta listagem, faça GOTO1. Use a listagem da fig. 4 para criar os caracteres desejados, dando entrada de um número seguido de NEWLINE por cada 15 entradas do caracter. Repare que o caracter vai sendo construído enquanto é dada cada entrada. Quando terminar todas as entradas, saia da rotina, fazendo NEWLINE. Pode ir passando as diferentes opções por sucessivas aplicações do comando NEWLINE. Accione a tecla «S» para saída do programa; se o quiser gravar, use GOTO 620.

### COMO UTILIZAR O PROGRAMA

Assegure-se que a RAMTOP se encontra em 32 000. O programa entrará em funcionamento automaticamente logo que carregado.

Se por acaso for necessário iniciar o programa, faça GOTO 1 e, para responder à pergunta «QUALQUER CARACTER A CRIAR?», faça apenas NEWLINE.

Movimente o caracter até à posição desejada e fixe-o nesse ponto com as teclas previamente determinadas. O próximo caracter seleccionado aparecerá inicialmente sobreposto ao anterior, o que não significa erro pois pode deslocá-lo para a posição desejada.

A saída da rotina continua a ser NEWLINE. Se pretender os gráficos em Inverse-Vídeo, accione a tecla «S» quando for interrogado.

A figura 5 apresenta os caracteres já definidos. Para chamar os caracteres presentes no écran para um novo programa, faça NEWLINE e dê entrada do BASIC apresentado na fig. 6. Faça RUN ao programa e, quando aparecer a mensagem 0/60, as linhas do programa podem ser apagadas.

Os gráficos são apresentados, quer pelo comando directo PRINT Z\$, quer pelo comando numa linha de programa:

```
10 PRINT Z$, seguido de GOTO 10.
```

### CÓDIGO MÁQUINA

As rotinas de Código Máquina tem os seguintes endereços:

```
16529 C.UP
16541 PRINT
16591 C.DN
16604 INVRT
```



As rotinas C.UP e C.DN usam a instrução LDIR para transferir o bloco de 727 bytes do endereço do «display file» para o endereço 32 000 e vice-versa.

As rotinas PRINT funcionam da seguinte forma: os registos A e B contêm X e Y (deslocamento do referencial) — écran 0,0. Os «steps» (espaços) 19 e 20 incrementam o endereço do «display file» (variável do sistema) de um valor igual ao que está no registo B.

Os «steps» 25 e 26 incrementam o endereço do «display file» de um valor igual a 33 vezes o valor do registo A.

Tendo estabelecido a posição do PRINT, os «steps» 31 a 34 fazem o PRINT dos primeiros 3 bytes dos caracteres aumentados.

Os «steps» 35 a 37 incrementam o endereço do «display file» em 30, indicando a posição correspondente ao PRINT da próxima linha, sendo aí apresentados os 3 bytes seguintes. Este processo é repetido 5 vezes para um carácter completo. De cada vez que o registo C, previamente carregado com cinco no «step» 27, tenha sido incrementado em 0 e regressa ao BASIC.

A rotina INVRT encontra cada carácter no «display file», acrescenta 128 ao valor presente e limpa o valor anterior.

LISTAGEM 1 — O programa de «carregamento» em Hex.

```

1 REM .....1.....2...
...3.....4.....5.....
...6.....7.....8.....
...9.....10.....
10 LET X=16514
20 INPUT A$
30 IF A$="" OR INT (LEN A$/2)
/LEN A$/2 THEN GO TO 20
40 PRINT AT 21,0;" "
T 21,1/A$
50 INPUT Z$
60 IF INT (LEN Z$/2) /LEN Z$/2
THEN GO TO 50
70 IF Z$="" THEN GO TO 100
80 PRINT AT 21,0;" "
90 GO TO 20
100 SCROLL
110 POKE 2,16+CODE A$+CODE A$/2
-476
120 LET Y=X+1
130 LET A$=A$(3 TO )
140 IF A$="" THEN GO TO 110
150 IF Z$="" THEN STOP
160 LET A$=Z$
170 GO TO 40

```

LISTAGEM 2 — O código Hex. — introduza cada linha de código da coluna esquerda e faça NEWLINE.

```

000000 LD HL,(16396) C.UP
11807D LD DE,32000
01D702 LD BC,727
ED80 LDIR
C9 RET
0680 LD B,0
03E0 LD A,0
79 PUSH AF
112100 LD DE,33
EA0C40 LD HL,(16396)
79 LD A,B
FE00 CP 0
0603 JRZ,+4
20 INC HL
10FD DJNZ,-3
F1 POP AF

```

```

FE00 CP 0
2804 JRZ,+4
47 LD B,A
10 ADD HL,DE
10FD DJNZ,-3
0605 LD C,5
118140 LD DE,16513
0603 LD B,3
13 INC DE
20 INC HL
10 LD A,(DE)
77 LD (HL),A
10FA DJNZ,-6
061E LD B,30
20 INC HL
10FD DJNZ,-3
00 DEC C
00F0 JANZ,-16
C9 RET
01007D LD HL,32000
ED80C40 LD DE,(16396)
01D702 LD BC,727
ED80 LDIR
C9 RET
EA0C40 LD HL,(16396)
061E LD B,31
0605 LD C,30
20 INC HL
7E LD A,(HL)
0680 ADD A,128
77 LD (HL),A
00 DEC C
00F0 JANZ,-8
20 INC HL
10FD DJNZ,-13
C9 RET

```

LISTAGEM 3 — A parte principal do programa em BASIC.

```

1 REM
2 NOT REM
3 GOSUB 100
4 RETURN
5 GOTO 100
6 GOTO 100
7 GOTO 100
8 GOTO 100
9 GOTO 100
10 GOTO 100
11 GOTO 100
12 GOTO 100
13 GOTO 100
14 GOTO 100
15 GOTO 100
16 GOTO 100
17 GOTO 100
18 GOTO 100
19 GOTO 100
20 GOTO 100
21 GOTO 100
22 GOTO 100
23 GOTO 100
24 GOTO 100
25 GOTO 100
26 GOTO 100
27 GOTO 100
28 GOTO 100
29 GOTO 100
30 GOTO 100
31 GOTO 100
32 GOTO 100
33 GOTO 100
34 GOTO 100
35 GOTO 100
36 GOTO 100
37 GOTO 100
38 GOTO 100
39 GOTO 100
40 GOTO 100
41 GOTO 100
42 GOTO 100
43 GOTO 100
44 GOTO 100
45 GOTO 100
46 GOTO 100
47 GOTO 100
48 GOTO 100
49 GOTO 100
50 GOTO 100
51 GOTO 100
52 GOTO 100
53 GOTO 100
54 GOTO 100
55 GOTO 100
56 GOTO 100
57 GOTO 100
58 GOTO 100
59 GOTO 100
60 GOTO 100
61 GOTO 100
62 GOTO 100
63 GOTO 100
64 GOTO 100
65 GOTO 100
66 GOTO 100
67 GOTO 100
68 GOTO 100
69 GOTO 100
70 GOTO 100
71 GOTO 100
72 GOTO 100
73 GOTO 100
74 GOTO 100
75 GOTO 100
76 GOTO 100
77 GOTO 100
78 GOTO 100
79 GOTO 100
80 GOTO 100
81 GOTO 100
82 GOTO 100
83 GOTO 100
84 GOTO 100
85 GOTO 100
86 GOTO 100
87 GOTO 100
88 GOTO 100
89 GOTO 100
90 GOTO 100
91 GOTO 100
92 GOTO 100
93 GOTO 100
94 GOTO 100
95 GOTO 100
96 GOTO 100
97 GOTO 100
98 GOTO 100
99 GOTO 100
100 GOTO 100
101 GOTO 100
102 GOTO 100
103 GOTO 100
104 GOTO 100
105 GOTO 100
106 GOTO 100
107 GOTO 100
108 GOTO 100
109 GOTO 100
110 GOTO 100
111 GOTO 100
112 GOTO 100
113 GOTO 100
114 GOTO 100
115 GOTO 100
116 GOTO 100
117 GOTO 100
118 GOTO 100
119 GOTO 100
120 GOTO 100
121 GOTO 100
122 GOTO 100
123 GOTO 100
124 GOTO 100
125 GOTO 100
126 GOTO 100
127 GOTO 100
128 GOTO 100
129 GOTO 100
130 GOTO 100
131 GOTO 100
132 GOTO 100
133 GOTO 100
134 GOTO 100
135 GOTO 100
136 GOTO 100
137 GOTO 100
138 GOTO 100
139 GOTO 100
140 GOTO 100
141 GOTO 100
142 GOTO 100
143 GOTO 100
144 GOTO 100
145 GOTO 100
146 GOTO 100
147 GOTO 100
148 GOTO 100
149 GOTO 100
150 GOTO 100
151 GOTO 100
152 GOTO 100
153 GOTO 100
154 GOTO 100
155 GOTO 100
156 GOTO 100
157 GOTO 100
158 GOTO 100
159 GOTO 100
160 GOTO 100
161 GOTO 100
162 GOTO 100
163 GOTO 100
164 GOTO 100
165 GOTO 100
166 GOTO 100
167 GOTO 100
168 GOTO 100
169 GOTO 100
170 GOTO 100
171 GOTO 100
172 GOTO 100
173 GOTO 100
174 GOTO 100
175 GOTO 100
176 GOTO 100
177 GOTO 100
178 GOTO 100
179 GOTO 100
180 GOTO 100
181 GOTO 100
182 GOTO 100
183 GOTO 100
184 GOTO 100
185 GOTO 100
186 GOTO 100
187 GOTO 100
188 GOTO 100
189 GOTO 100
190 GOTO 100
191 GOTO 100
192 GOTO 100
193 GOTO 100
194 GOTO 100
195 GOTO 100
196 GOTO 100
197 GOTO 100
198 GOTO 100
199 GOTO 100
200 GOTO 100
201 GOTO 100
202 GOTO 100
203 GOTO 100
204 GOTO 100
205 GOTO 100
206 GOTO 100
207 GOTO 100
208 GOTO 100
209 GOTO 100
210 GOTO 100
211 GOTO 100
212 GOTO 100
213 GOTO 100
214 GOTO 100
215 GOTO 100
216 GOTO 100
217 GOTO 100
218 GOTO 100
219 GOTO 100
220 GOTO 100
221 GOTO 100
222 GOTO 100
223 GOTO 100
224 GOTO 100
225 GOTO 100
226 GOTO 100
227 GOTO 100
228 GOTO 100
229 GOTO 100
230 GOTO 100
231 GOTO 100
232 GOTO 100
233 GOTO 100
234 GOTO 100
235 GOTO 100
236 GOTO 100
237 GOTO 100
238 GOTO 100
239 GOTO 100
240 GOTO 100
241 GOTO 100
242 GOTO 100
243 GOTO 100
244 GOTO 100
245 GOTO 100
246 GOTO 100
247 GOTO 100
248 GOTO 100
249 GOTO 100
250 GOTO 100
251 GOTO 100
252 GOTO 100
253 GOTO 100
254 GOTO 100
255 GOTO 100
256 GOTO 100
257 GOTO 100
258 GOTO 100
259 GOTO 100
260 GOTO 100
261 GOTO 100
262 GOTO 100
263 GOTO 100
264 GOTO 100
265 GOTO 100
266 GOTO 100
267 GOTO 100
268 GOTO 100
269 GOTO 100
270 GOTO 100
271 GOTO 100
272 GOTO 100
273 GOTO 100
274 GOTO 100
275 GOTO 100
276 GOTO 100
277 GOTO 100
278 GOTO 100
279 GOTO 100
280 GOTO 100
281 GOTO 100
282 GOTO 100
283 GOTO 100
284 GOTO 100
285 GOTO 100
286 GOTO 100
287 GOTO 100
288 GOTO 100
289 GOTO 100
290 GOTO 100
291 GOTO 100
292 GOTO 100
293 GOTO 100
294 GOTO 100
295 GOTO 100
296 GOTO 100
297 GOTO 100
298 GOTO 100
299 GOTO 100
300 GOTO 100
301 GOTO 100
302 GOTO 100
303 GOTO 100
304 GOTO 100
305 GOTO 100
306 GOTO 100
307 GOTO 100
308 GOTO 100
309 GOTO 100
310 GOTO 100
311 GOTO 100
312 GOTO 100
313 GOTO 100
314 GOTO 100
315 GOTO 100
316 GOTO 100
317 GOTO 100
318 GOTO 100
319 GOTO 100
320 GOTO 100
321 GOTO 100
322 GOTO 100
323 GOTO 100
324 GOTO 100
325 GOTO 100
326 GOTO 100
327 GOTO 100
328 GOTO 100
329 GOTO 100
330 GOTO 100
331 GOTO 100
332 GOTO 100
333 GOTO 100
334 GOTO 100
335 GOTO 100
336 GOTO 100
337 GOTO 100
338 GOTO 100
339 GOTO 100
340 GOTO 100
341 GOTO 100
342 GOTO 100
343 GOTO 100
344 GOTO 100
345 GOTO 100
346 GOTO 100
347 GOTO 100
348 GOTO 100
349 GOTO 100
350 GOTO 100
351 GOTO 100
352 GOTO 100
353 GOTO 100
354 GOTO 100
355 GOTO 100
356 GOTO 100
357 GOTO 100
358 GOTO 100
359 GOTO 100
360 GOTO 100
361 GOTO 100
362 GOTO 100
363 GOTO 100
364 GOTO 100
365 GOTO 100
366 GOTO 100
367 GOTO 100
368 GOTO 100
369 GOTO 100
370 GOTO 100
371 GOTO 100
372 GOTO 100
373 GOTO 100
374 GOTO 100
375 GOTO 100
376 GOTO 100
377 GOTO 100
378 GOTO 100
379 GOTO 100
380 GOTO 100
381 GOTO 100
382 GOTO 100
383 GOTO 100
384 GOTO 100
385 GOTO 100
386 GOTO 100
387 GOTO 100
388 GOTO 100
389 GOTO 100
390 GOTO 100
391 GOTO 100
392 GOTO 100
393 GOTO 100
394 GOTO 100
395 GOTO 100
396 GOTO 100
397 GOTO 100
398 GOTO 100
399 GOTO 100
400 GOTO 100
401 GOTO 100
402 GOTO 100
403 GOTO 100
404 GOTO 100
405 GOTO 100
406 GOTO 100
407 GOTO 100
408 GOTO 100
409 GOTO 100
410 GOTO 100
411 GOTO 100
412 GOTO 100
413 GOTO 100
414 GOTO 100
415 GOTO 100
416 GOTO 100
417 GOTO 100
418 GOTO 100
419 GOTO 100
420 GOTO 100
421 GOTO 100
422 GOTO 100
423 GOTO 100
424 GOTO 100
425 GOTO 100
426 GOTO 100
427 GOTO 100
428 GOTO 100
429 GOTO 100
430 GOTO 100
431 GOTO 100
432 GOTO 100
433 GOTO 100
434 GOTO 100
435 GOTO 100
436 GOTO 100
437 GOTO 100
438 GOTO 100
439 GOTO 100
440 GOTO 100
441 GOTO 100
442 GOTO 100
443 GOTO 100
444 GOTO 100
445 GOTO 100
446 GOTO 100
447 GOTO 100
448 GOTO 100
449 GOTO 100
450 GOTO 100
451 GOTO 100
452 GOTO 100
453 GOTO 100
454 GOTO 100
455 GOTO 100
456 GOTO 100
457 GOTO 100
458 GOTO 100
459 GOTO 100
460 GOTO 100
461 GOTO 100
462 GOTO 100
463 GOTO 100
464 GOTO 100
465 GOTO 100
466 GOTO 100
467 GOTO 100
468 GOTO 100
469 GOTO 100
470 GOTO 100
471 GOTO 100
472 GOTO 100
473 GOTO 100
474 GOTO 100
475 GOTO 100
476 GOTO 100
477 GOTO 100
478 GOTO 100
479 GOTO 100
480 GOTO 100
481 GOTO 100
482 GOTO 100
483 GOTO 100
484 GOTO 100
485 GOTO 100
486 GOTO 100
487 GOTO 100
488 GOTO 100
489 GOTO 100
490 GOTO 100
491 GOTO 100
492 GOTO 100
493 GOTO 100
494 GOTO 100
495 GOTO 100
496 GOTO 100
497 GOTO 100
498 GOTO 100
499 GOTO 100
500 GOTO 100
501 GOTO 100
502 GOTO 100
503 GOTO 100
504 GOTO 100
505 GOTO 100
506 GOTO 100
507 GOTO 100
508 GOTO 100
509 GOTO 100
510 GOTO 100
511 GOTO 100
512 GOTO 100
513 GOTO 100
514 GOTO 100
515 GOTO 100
516 GOTO 100
517 GOTO 100
518 GOTO 100
519 GOTO 100
520 GOTO 100
521 GOTO 100
522 GOTO 100
523 GOTO 100
524 GOTO 100
525 GOTO 100
526 GOTO 100
527 GOTO 100
528 GOTO 100
529 GOTO 100
530 GOTO 100
531 GOTO 100
532 GOTO 100
533 GOTO 100
534 GOTO 100
535 GOTO 100
536 GOTO 100
537 GOTO 100
538 GOTO 100
539 GOTO 100
540 GOTO 100
541 GOTO 100
542 GOTO 100
543 GOTO 100
544 GOTO 100
545 GOTO 100
546 GOTO 100
547 GOTO 100
548 GOTO 100
549 GOTO 100
550 GOTO 100
551 GOTO 100
552 GOTO 100
553 GOTO 100
554 GOTO 100
555 GOTO 100
556 GOTO 100
557 GOTO 100
558 GOTO 100
559 GOTO 100
560 GOTO 100
561 GOTO 100
562 GOTO 100
563 GOTO 100
564 GOTO 100
565 GOTO 100
566 GOTO 100
567 GOTO 100
568 GOTO 100
569 GOTO 100
570 GOTO 100
571 GOTO 100
572 GOTO 100
573 GOTO 100
574 GOTO 100
575 GOTO 100
576 GOTO 100
577 GOTO 100
578 GOTO 100
579 GOTO 100
580 GOTO 100
581 GOTO 100
582 GOTO 100
583 GOTO 100
584 GOTO 100
585 GOTO 100
586 GOTO 100
587 GOTO 100
588 GOTO 100
589 GOTO 100
590 GOTO 100
591 GOTO 100
592 GOTO 100
593 GOTO 100
594 GOTO 100
595 GOTO 100
596 GOTO 100
597 GOTO 100
598 GOTO 100
599 GOTO 100
600 GOTO 100
601 GOTO 100
602 GOTO 100
603 GOTO 100
604 GOTO 100
605 GOTO 100
606 GOTO 100
607 GOTO 100
608 GOTO 100
609 GOTO 100
610 GOTO 100
611 GOTO 100
612 GOTO 100
613 GOTO 100
614 GOTO 100
615 GOTO 100
616 GOTO 100
617 GOTO 100
618 GOTO 100
619 GOTO 100
620 GOTO 100
621 GOTO 100
622 GOTO 100
623 GOTO 100
624 GOTO 100
625 GOTO 100
626 GOTO 100
627 GOTO 100
628 GOTO 100
629 GOTO 100
630 GOTO 100
631 GOTO 100
632 GOTO 100
633 GOTO 100
634 GOTO 100
635 GOTO 100
636 GOTO 100
637 GOTO 100
638 GOTO 100
639 GOTO 100
640 GOTO 100
641 GOTO 100
642 GOTO 100
643 GOTO 100
644 GOTO 100
645 GOTO 100
646 GOTO 100
647 GOTO 100
648 GOTO 100
649 GOTO 100
650 GOTO 100
651 GOTO 100
652 GOTO 100
653 GOTO 100
654 GOTO 100
655 GOTO 100
656 GOTO 100
657 GOTO 100
658 GOTO 100
659 GOTO 100
660 GOTO 100
661 GOTO 100
662 GOTO 100
663 GOTO 100
664 GOTO 100
665 GOTO 100
666 GOTO 100
667 GOTO 100
668 GOTO 100
669 GOTO 100
670 GOTO 100
671 GOTO 100
672 GOTO 100
673 GOTO 100
674 GOTO 100
675 GOTO 100
676 GOTO 100
677 GOTO 100
678 GOTO 100
679 GOTO 100
680 GOTO 100
681 GOTO 100
682 GOTO 100
683 GOTO 100
684 GOTO 100
685 GOTO 100
686 GOTO 100
687 GOTO 100
688 GOTO 100
689 GOTO 100
690 GOTO 100
691 GOTO 100
692 GOTO 100
693 GOTO 100
694 GOTO 100
695 GOTO 100
696 GOTO 100
697 GOTO 100
698 GOTO 100
699 GOTO 100
700 GOTO 100
701 GOTO 100
702 GOTO 100
703 GOTO 100
704 GOTO 100
705 GOTO 100
706 GOTO 100
707 GOTO 100
708 GOTO 100
709 GOTO 100
710 GOTO 100
711 GOTO 100
712 GOTO 100
713 GOTO 100
714 GOTO 100
715 GOTO 100
716 GOTO 100
717 GOTO 100
718 GOTO 100
719 GOTO 100
720 GOTO 100
721 GOTO 100
722 GOTO 100
723 GOTO 100
724 GOTO 100
725 GOTO 100
726 GOTO 100
727 GOTO 100
728 GOTO 100
729 GOTO 100
730 GOTO 100
731 GOTO 100
732 GOTO 100
733 GOTO 100
734 GOTO 100
735 GOTO 100
736 GOTO 100
737 GOTO 100
738 GOTO 100
739 GOTO 100
740 GOTO 100
741 GOTO 100
742 GOTO 100
743 GOTO 100
744 GOTO 100
745 GOTO 100
746 GOTO 100
747 GOTO 100
748 GOTO 100
749 GOTO 100
750 GOTO 100
751 GOTO 100
752 GOTO 100
753 GOTO 100
754 GOTO 100
755 GOTO 100
756 GOTO 100
757 GOTO 100
758 GOTO 100
759 GOTO 100
760 GOTO 100
761 GOTO 100
762 GOTO 100
763 GOTO 100
764 GOTO 100
765 GOTO 100
766 GOTO 100
767 GOTO 100
768 GOTO 100
769 GOTO 100
770 GOTO 100
771 GOTO 100
772 GOTO 100
773 GOTO 100
774 GOTO 100
775 GOTO 100
776 GOTO 100
777 GOTO 100
778 GOTO 100
779 GOTO 100
780 GOTO 100
781 GOTO 100
782 GOTO 100
783 GOTO 100
784 GOTO 100
785 GOTO 100
786 GOTO 100
787 GOTO 100
788 GOTO 100
789 GOTO 100
790 GOTO 100
791 GOTO 100
792 GOTO 100
793 GOTO 100
794 GOTO 100
795 GOTO 100
796 GOTO 100
797 GOTO 100
798 GOTO 100
799 GOTO 100
800 GOTO 100
801 GOTO 100
802 GOTO 100
803 GOTO 100
804 GOTO 100
805 GOTO 100
806 GOTO 100
807 GOTO 100
808 GOTO 100
809 GOTO 100
810 GOTO 100
811 GOTO 100
812 GOTO 100
813 GOTO 100
814 GOTO 100
815 GOTO 100
816 GOTO 100
817 GOTO 100
818 GOTO 100
819 GOTO 100
820 GOTO 100
821 GOTO 100
822 GOTO 100
823 GOTO 100
824 GOTO 100
825 GOTO 100
826 GOTO 100
827 GOTO 100
828 GOTO 100
829 GOTO 100
830 GOTO 100
831 GOTO 100
832 GOTO 100
833 GOTO 100
834 GOTO 100
835 GOTO 100
836 GOTO 100
837 GOTO 100
838 GOTO 100
839 GOTO 100
840 GOTO 100
841 GOTO 100
842 GOTO 100
843 GOTO 100
844 GOTO 100
845 GOTO 100
846 GOTO 100
847 GOTO 100
848 GOTO 100
849 GOTO 100
850 GOTO 100
851 GOTO 100
852 GOTO 100
853 GOTO 100
854 GOTO 100
855 GOTO 100
856 GOTO 100
857 GOTO 100
858 GOTO 100
859 GOTO 100
860 GOTO 100
861 GOTO 100
862 GOTO 100
863 GOTO 100
864 GOTO 100
865 GOTO 100
866 GOTO 100
867 GOTO 100
868 GOTO 100
869 GOTO 100
870 GOTO 100
871 GOTO 100
872 GOTO 100
873 GOTO 100
874 GOTO 100
875 GOTO 100
876 GOTO 100
877 GOTO 100
878 GOTO 100
879 GOTO 100
880 GOTO 100
881 GOTO 100
882 GOTO 100
883 GOTO 100
884 GOTO 100
885 GOTO 100
886 GOTO 100
887 GOTO 100
888 GOTO 100
889 GOTO 100
890 GOTO 100
891 GOTO 100
892 GOTO 100
893 GOTO 100
894 GOTO 100
895 GOTO 100
896 GOTO 100
897 GOTO 100
898 GOTO 100
899 GOTO 100
900 GOTO 100
901 GOTO 100
902 GOTO 100
903 GOTO 100
904 GOTO 100
905 GOTO 100
906 GOTO 100
907 GOTO 100
908 GOTO 100
909 GOTO 100
910 GOTO 100
911 GOTO 100
912 GOTO 100
913 GOTO 100
914 GOTO 100
915 GOTO 100
916 GOTO 100
917 GOTO 100
918 GOTO 100
919 GOTO 100
920 GOTO 100
921 GOTO 100
922 GOTO 100
923 GOTO 100
924 GOTO 100
925 GOTO 100
926 GOTO 100
927 GOTO 100
928 GOTO 100
929 GOTO 100
930 GOTO 100
931 GOTO 100
932 GOTO 100
933 GOTO 100
934 GOTO 100
935 GOTO 100
936 GOTO 100
937 GOTO 100
938 GOTO 100
939 GOTO 100
940 GOTO 100
941 GOTO 100
942 GOTO 100
943 GOTO 100
944 GOTO 100
945 GOTO 100
946 GOTO 100
947 GOTO 100
948 GOTO 100
949 GOTO 100
950 GOTO 100
951 GOTO 100
952 GOTO 100
953 GOTO 100
954 GOTO 100
955 GOTO 100
956 GOTO 100
957 GOTO 100
958 GOTO 100
959 GOTO 100
960 GOTO 100
961 GOTO 100
962 GOTO 100
963 GOTO 100
964 GOTO 100
965 GOTO 100
966 GOTO 100
967 GOTO 100
968 GOTO 100
969 GOTO 100
970 GOTO 100
971 GOTO 100
972 GOTO 100
973 GOTO 100
974 GOTO 100
975 GOTO 100
976 GOTO 100
977 GOTO 100
978 GOTO 100
979 GOTO 100
980 GOTO 100
981 GOTO 100
982 GOTO 100
983 GOTO 100
984 GOTO 100
985 GOTO 100
986 GOTO 100
987 GOTO 100
988 GOTO 100
989 GOTO 100
990 GOTO 100
991 GOTO 100
992 GOTO 100
993 GOTO 100
994 GOTO 100
995 GOTO 100
996 GOTO 100
997 GOTO 100
998 GOTO 100
999 GOTO 100
1000 GOTO 100

```

```

PROCTER
000000 GOSUB 640
000000 LET Z=100014
000000 FOR M=1 TO 5
000000 FOR N=1 TO 5
000000 POKE M, CODE R#(C,M,N)
000000 LET I=N-1
000000 NEXT N
000000 NEXT M
000000 PRINT AT 21.0, "MOVA S.S.T.E
000000 CONFIRMA"
000000 RAND USR 100000
000000 RAND USR 100000
000000 IF INKEYS="0" AND X<0 THEN
000000 X=X-1
000000 IF INKEYS="6" AND X<0 THEN
000000 X=X+1
000000 IF INKEYS="7" AND X<0 THEN
000000 Y=Y-1
000000 IF INKEYS="8" AND X<0 THEN
000000 Y=Y+1
000000 IF INKEYS="9" THEN GOTO 660
000000 POKE 100014,X
000000 POKE 100014,Y
000000 RAND USR 100000
000000 GOTO 620
000000 REM -----INTE
000000 PRINT AT 21.0, "INVESTE
000000 (S)
000000 INPUT C$
000000 IF C$="0" THEN GOTO 670
000000 RAND USR 100000
000000 RAND USR 100000
000000 PRINT AT 21.0, "SAIDA DO JO
000000 G
000000 INPUT C$
000000 IF C$="0" THEN GOTO 680
000000 STOP
000000 GOTO 680
000000 LET C=CODE C$-10
000000 IF C=100 THEN LET C=1
000000 IF C=10 THEN LET C=1
000000 RETURN

```

LISTAGEM 4 — Desta lista, escolha o(s) conjunto(s) de 15 entradas necessá-  
rias à criação do(s) seu(s) caractere(s). Depois faça  
NEWLINE.

" "	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	135	0	4	0	0	0	0
"5"	0	133	0	0	0	0	0
"6"	135	130	0	0	0	1	0
"7"	1	134	130	1	0	1	0
"8"	0	0	0	0	0	0	1
"9"	0	0	0	0	0	1	0
"0"	0	0	4	1	0	0	0
"1"	0	0	1	0	0	1	0
"2"	135	1	0	0	0	0	0
"3"	0	134	0	0	0	1	0
"4"	134	0	0	0	0	0	0
"5"	0	135	1	0	0	0	0
"6"	0	0	0	134	0	0	135
"7"	1	0	1	0	0	0	0
"8"	0	0	0	0	135	1	0
"9"	0	0	1	0	0	0	0
"0"	0	0	0	131	131	4	131
131	4	0	0	0	0	0	0

"1"	0	0	0	0	0	0	0
"2"	1	0	1	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	1	0	0	0	0	0	0
"5"	0	4	0	134	130	1	130
130	1	1	0	1	0	0	0
"6"	0	0	0	0	133	0	135
"7"	1	0	0	0	0	1	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	1	0
"5"	0	0	0	0	0	0	0
"6"	0	0	4	0	130	0	130
"7"	0	0	0	0	0	0	0
"8"	130	0	0	130	0	0	130
"9"	0	0	0	0	0	1	0
"0"	0	0	0	0	0	0	135
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	130	0	0	130	1	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0	0	0	0
"3"	0	0	0	0	0	0	0
"4"	0	0	0	0	0	0	0
"5"	0	0	0	0	0	0	0
"6"	0	0	0	0	0	0	0
"7"	0	0	0	0	0	0	0
"8"	0	0	0	0	0	0	0
"9"	0	0	0	0	0	0	0
"0"	0	0	0	0	0	0	0
"1"	0	0	0	0	0	0	0
"2"	0	0	0	0			



"X"	0	0	0	0	0	0	7
4	0	0	0	0	0	0	1
"L"	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1
"H"	7	0	0	0	0	0	0
0	0	0	0	0	1	0	1
"N"	0	0	0	7	4	0	0
0	0	0	0	0	1	0	1
"O"	0	0	0	4	0	0	0
0	0	0	0	0	0	0	0
"P"	7	0	0	4	0	0	7
0	0	0	0	0	1	0	0
"Q"	0	0	0	4	0	0	0
0	0	0	134	1	0	1	1
"R"	7	0	0	4	0	0	7
0	0	0	0	4	1	0	1
"S"	0	0	0	4	0	0	0
0	4	4	0	0	0	0	0
"T"	0	0	7	1	0	0	0
0	0	0	0	0	0	1	0
"U"	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
"V"	0	0	0	0	0	0	0
0	0	134	135	1	0	1	0
"W"	0	0	0	0	0	0	0
4	0	0	0	0	0	0	1

"X"	0	0	0	0	0	0	0
0	0	0	0	4	1	0	1
"Y"	0	0	0	0	134	135	1
0	0	0	0	0	0	1	0
"Z"	0	0	0	0	135	1	135
1	0	0	0	0	0	0	1

FIGURA 5 — Os caracteres criados pelo programa.

1234567890  
 ABCDEFGHIJK  
 LMNOPQRSTU  
 VWXYZ\*£(€/?

LISTAGEM 6 — Utilize este programa para introduzir em qualquer programa (BASIC) os caracteres já definidos.

```

10 FAST
20 LET Z$=""
30 FOR X=32000 TO 32692
40 IF PEEK X<>118 THEN LET Z$=
  Z$+CHR PEEK X
50 NEXT X
60 PRINT Z$
70 END

```

## ANIMAÇÃO NO SPECTRUM

16 e 48 K

In. YOUR COMPUTER, Abril 1983

Trad. e Adapt.: J. MAGALHÃES

A possibilidade de cor, som e capacidade de gráficos de alta redução no SPECTRUM encorajou muitas pessoas a escrever programas de jogos. É claro que a maioria deles exige bastante velocidade para maior entusiasmo.

O método mais utilizado para produzir ilusão de movimento começa por uma posição inicial do objecto, seguindo-se uma nova posição mais à frente e o desaparecimento da anterior. Por vezes este método é suficiente para um programa em BASIC, com o inconveniente de se aperceber um tipo de «empurrão» à figura inicial.

Dentro do software comercial, utiliza-se o código máquina para se obter um movimento mais real e mais rápido. Ora, isto não está acessível a muitas pessoas que programem para si mesmas em suas casas. Será necessário considerar outros métodos de se conseguir a velocidade em programas BASIC.

Por isso, vamos apresentar uma pequena rotina em código máquina que, esperamos, satisfaça as suas necessidades. Esta rotina foi aperfeiçoada de modo a poder utilizar gráficos («user-defined») de qualquer tamanho, sendo chamados sempre que pretendidos. Podem-se deslocar em qualquer posição, num écran de alta resolução 256x176, o que já é considerável em relação ao écran 32x22 (o écran normal). O código máquina ocupa 197 bytes que podem ser guardados deslocando a RAMTOP na zona mais abaixo da memó-

ria, usando para isso o comando CLEAR e colocando o código entre a área BASIC e a área de Gráficos.

NOTA: Os endereços diferem no 48K e no 16K sendo, no entanto, a mesma rotina. O bloco de bytes do código máquina encontra-se na listagem 1.

Os possuidores do SPECTRUM 48K podem usar o programa de carregamento da listagem 2. Dê entrada deste programa e faça RUN; introduza os bytes um a um da esquerda para a direita. Antes de fazer a gravação (depois de ter dado todas as entradas), sugerimos que confira os dados retirando do programa (listagem 2) as linhas 20, 30, 70 e 90 e substituindo a linha 80 por:

```
80 PRINT TAB 10; PEEK i
```

Se já conferiu, então grave da seguinte forma:

```
SAVE «animate» CODE 65160, 197
```

No caso do SPECTRUM 16K, serão necessárias as seguintes alterações na listagem 2:

— Linha 20 será CLEAR 32389

— Linha 50 será FOR i = 32390 TO 32586.



Para gravar:

SAVE «animate» CODE 32390, 197

Agora pode testar a rotina, limpando o écran (CLS; ENTER) e usando o comando:

RANDOMIZE USR 65171 — SPECTRUM 48K  
RANDOMIZE USR 32401 — SPECTRUM 16K

No canto superior esquerdo do écran deverá aparecer o carácter «A». Se isso não acontecer é porque houve qualquer engano na introdução do código máquina.

Se o programa «desaparecer», recarregue-o (previamente gravado) sem esquecer limpar a RAMTOP pelo CLEAR apropriado. Verifique.

Se tudo correu tal como indicámos, passa-se ao programa BASIC.

Repare na listagem 3 (rotina «disassembled») que os primeiros 6 bytes são usados como variáveis XPOS, YPOS, etc.. Estes nomes não serão reconhecidos se tentar usá-los no programa BASIC; mas, fazendo diferentes valores para o POKE nestes 6 bytes, pode controlar a operação da rotina como se segue.

#### CONTROLO DA ROTINA

- XPOS — endereço 32390 para o SPECTRUM 16K  
— endereço 65160 para o SPECTRUM 48K

«X» é a coordenada da posição no canto superior esquerdo do écran (posição inicial do gráfico). São possíveis outros valores como para o comando PLOT de 0 a 255.

- YPOS — endereço 32391 (SPECTRUM 16K)  
— endereço 65161 (SPECTRUM 48K)

«Y» é a coordenada da posição inicial do gráfico, no canto superior esquerdo do écran. Pode tomar valores de 0 a 175 (topo do écran).

- EXTENSÃO («width») — endereço 32392 (16K)  
— endereço 65162 (48K)

Representa o número de pontos da esquerda para a direita.

- HT — endereço 32393 (16 K)  
— endereço 65163 (48 K)

Número de pontos do fundo ao topo do écran.

- MODE — endereço 32394 (16 K)  
— endereço 65164 (48 K)

Toma os valores 1 se deseja o carácter na presente posição, ou 0 se deseja limpar um bloco de extensão × pontos HT na presente posição.

- UDGCH — endereço 32395 (16 K)  
— endereço 65165 (48 K)

Deve tomar valores entre 1 a 21, que transmite à rotina onde encontrar os elementos DATA do gráfico que deseja reproduzir. O valor 1 significa que o 1.º byte de DATA se encontra no

endereço USR «a», que é o 1.º byte do gráfico «a». O valor 21 significa que se encontra no endereço USR «u».

Os valores POKE nestes 6 bytes, antes de chamada a rotina, serão alterados à saída. Por isso, se utilizar mais de um carácter não precisa renovar («update») todos os bytes.

Terá resultados bastante estranhos ou pode até perder o programa se pretende valores para XPOS e YPOS que não permitam a posição no espaço que compreende o écran. Então o seu programa BASIC precisa prever este caso.

#### DIFERENTES CORES

A rotina apresenta o carácter gráfico com a cor definida pelos dados do sistema variável ATTR T — localização 23695. Se pretender cores diferentes pode usar ATTR T a partir do BASIC, sem alterar a presente posição, por uma simulação da instrução PRINT:

PRINT PAPER 6; INK 1;

Antes de ser chamada a rotina. Assim, podem ser usados os comandos FLASH e BRIGHT.

Se usar a rotina para gráficos de 8 por 8 pontos («user-graphics»), o método de armazenamento é exactamente igual ao descrito no capítulo 14 do manual do SPECTRUM. Contudo, a rotina pode produzir gráficos de qualquer dimensão, mas através de um método diferente.

Inicialmente, desenhe o carácter numa folha de papel quadriculado. Dado que vamos usar a rotina para mover o carácter de 2 em 2 pontos, é aconselhável deixar uma margem de duas colunas vazias à esquerda e à direita. A operação que faz mover o carácter da esquerda para a direita em 2 pontos, apresentando uma nova posição, irá automaticamente cobrir o anterior «invasor» (o 1.º carácter), evitando erros. O carácter que queremos usar é de 14 por 11 pontos.

Se a extensão não for rigorosamente múltipla de 8, deverá acrescentar o número de colunas vazias necessárias do lado direito. A extensão do carácter é então dividida em oito pequenas secções que podem ser descritas pelo vulgar número de BIN e armazenadas na área de gráficos «user-defined»). Para isso, pode usar-se um programa como o da listagem 4. O 1.º byte é armazenado no endereço dado por USR «a». Aquando da entrada de dados para os seus caracteres, lembre-se que DATA para a 1.ª linha é armazenado em primeiro lugar, depois a 2.ª linha, e assim sucessivamente. Desde que o «invasor» ocupe 22 bytes (isto é, todos os gráficos «a», «b» e a maior parte de «c»), a secção seguinte não usada começará em USR «d», onde podem ser armazenados os 4 bytes de DATA necessários para o carácter da figura 2. No invasor, as pernas aparecerão em movimento enquanto ele se move no écran. Dê entrada e faça RUN da listagem 4. Se quiser, grave estes gráficos:

SAVE «chars» CODE USR «a», 32

Pode limpar o écran com NEW, desde que a rotina de gráficos se encontre na parte superior da RAMTOP — e introduza o programa da listagem 5. Aparecerá uma linha de 7 invasores multicolores, movimentando-se tal como no tradicional jogo «INVADERS».

NOTA: Os invasores podem ser apagados se for chamada a rotina quando eles se movem uma linha abaixo, bastando para isso alterar **Mode** para 0 (zero), **Extensão** para 140 e **HT** para 11 (linhas 130-160).



Telef. 25970 — 3500 VISEU



## Listagem 3 — Código-Máquina "disassembled"

```

      ORG 65160
XPOS DEFB 2
YPOS DEFB 175
WIDTH DEFB 8
HT DEFB 8
MODE DEFB 1
UDGCH DEFB 1
      DEFB 0
      DEFB 0
      DEFB 0
      DEFB 0
      DEFB 0
;
START LD IX, (23730)
      INC IX
      LD HL, (23675)
      LD E, (IX+5)      LN1
      DEC E      L1
      SLA E
      SLA E
      SLA E
      LD D, 0      L2
      ADD HL, DE
      EX HL, DE
      LD C, (IX+0)
      LD B, (IX+1)      L3
      LD A, (IX+3)
LINE  PUSH AF
      PUSH BC
      LD A, (IX+2)
      LD (IX+6), A
      CALL 22AAH
      LD (IX+7), A
      INC (IX+7)      L5
      CPL
      AND 7
      INC A
      LD (IX+8), A
      PUSH DE
      PUSH HL
      CALL 0BDBH
      POP HL
      POP DE
      BIT 0, (IX+4)      L7
      JR NZ, 5
      LD BC, 0
      JR LN1
      EX HL, DE
      LD B, (HL)
      INC HL
      LD C, (HL)
      EX HL, DE      L4
      LD (IX+9), 8
      LD (IX+10), 9
      LD A, (HL)
      DEC (IX+7)      L9
      JR Z, L3
      RLCA
      DEC (IX+10)
      DEC (IX+7)      L8
      JR NZ, L2
      SLA C
      RL B
      RLA
      DEC (IX+10)
      DEC (IX+6)
      JR NZ, L7
      DEC (IX+10)
      JR Z, L6
      RLCA
      DEC (IX+10)
      JR NZ, L5
      LD (HL), A
      POP BC
      POP AF
      DEC B
      DEC A
      JR NZ, LINE
      RET
      DEC (IX+9)
      JR NZ, L4
      BIT 0, (IX+4)
      JR Z, L4
      EX HL, DE
      INC HL
      LD C, (HL)
      EX HL, DE
      DEC (IX+8)
      JR NZ, L3
      DEC (IX+10)
      JR Z, L8
      RLCA
      DEC (IX+10)
      JR NZ, L9
      JR NZ, L3
      LD (HL), A
      INC HL
      PUSH DE
      PUSH HL
      PUSH AF
      CALL 0BDBH
      POP AF
      POP HL
      POP DE
      LD (IX+8), 8
      LD (IX+7), 1
      JR L1
      END

```

## Listagem 4

```

10 REM chrs fig.1
20 FOR i=USR "a" TO USR "z" -21
30 READ d: POKE i,d
40 NEXT i
50 DATA BIN 00000111, BIN 10000
000, BIN 00011111, BIN 11100000, BI
N 00111111, BIN 11110000, BIN 0011
0011, BIN 00110000
60 DATA BIN 00110011, BIN 00110
000, BIN 00111111, BIN 11110000, BI
N 00011111, BIN 11100000, BIN 0001
0100, BIN 10100000
70 DATA BIN 00010011, BIN 00100
000, BIN 00100000, BIN 00010000, BI
N 00100000, BIN 00010000
80 REM chrs fig.2
90 FOR i=USR "d" TO USR "d"+3
100 READ d: POKE i,d
110 NEXT i
120 DATA BIN 00001000, BIN 01000
000, BIN 00001000, BIN 01000000

```

## Listagem 5

```

4 REM -----
5 CLEAR 65160
10 PAPER 7: BORDER 7: CLS
20 FOR y=160 TO 100 STEP -20
30 GO SUB 100
40 NEXT y
50 INK 0
60 STOP

```

```

70 REM *****
100 POKE 65162,14: POKE 65164,1
110 FOR x=0 TO 100 STEP 2: GO 5
UB 200: NEXT x
120 FOR x=100 TO 0 STEP -2: GO
SUB 200: NEXT x
130 POKE 65160,0: POKE 65161,0
140 POKE 65162,140: POKE 65163,
11
150 POKE 65164,0
160 RANDOMIZE USR 65171
170 RETURN
180 REM *****
200 POKE 65161,0: POKE 65163,11
210 POKE 65165,1
220 GO SUB 300
230 POKE 65161,4-9: POKE 65163,
2
240 POKE 65165,4
250 GO SUB 300
260 RETURN
270 REM *****
300 FOR c=0 TO 6: PRINT INK c
310 POKE 65160,c+20+x
320 RANDOMIZE USR 65171
330 NEXT c
340 RETURN
350 REM *****

```

Listagem 6 — Para o SPECTRUM 16 K, alterar a Linha 10:  
10 LET a=32390

```

10 LET a=65160
20 POKE a+158,24

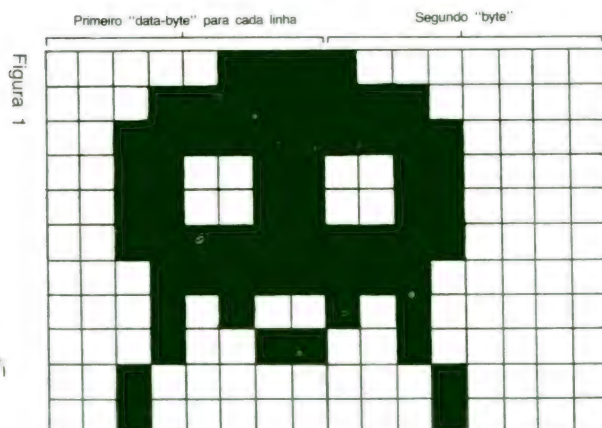
```



```

30 POKE a+159,37
40 FOR b=a+197 TO a+204
50 READ c: POKE b,c
60 NEXT b
70 DATA 78,205,221,64,9,3,24,2
11 80 SAVE "animate"CODE a,205

```



## PERSEGUIÇÃO

SPECTRUM 16 K

In. HOME COMPUTING

Trad. e Adapt.: ANTÔNIO AMARAL/Porto

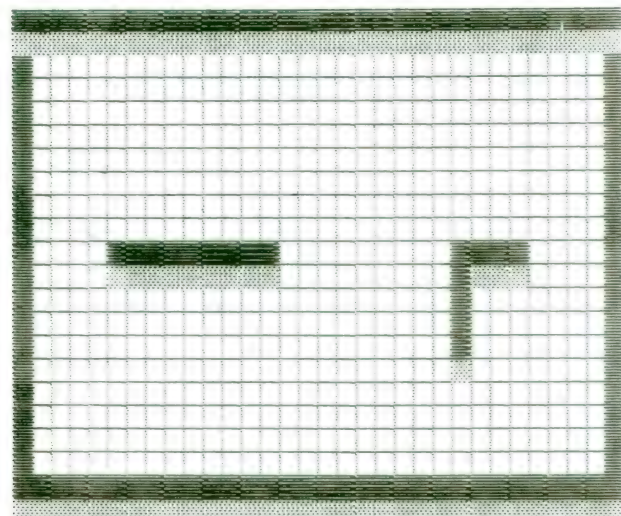
O objectivo da PERSEGUIÇÃO é empurrar o outro jogador para uma parede.

Cada jogador deixa uma parede e não pode parar até que um seja destruído.

### CONTROLES

Jogador 1: E D M A

Jogador 2: Q O P L



```

1 REM PERSEGUIÇÃO
2 GO SUB 9000
3 GO SUB 8000
4 LET s1=0: LET s2=0
5 PRINT AT 0,0;" "
6 PRINT AT 20,0;" "
7 PRINT AT 21,0;" "
8 PRINT AT 1,0;" "
9 FOR f=2 TO 19: PRINT AT f,0;" "
10 NEXT f
11 LET x=4: LET y=10
12 LET a=27: LET b=10
13 LET d=2: LET c=4
14 GO TO 100+60*(AND(,5)
15 LET a$="1" AND IN 63436(2
55)+("2" AND IN 64510=253)+("3"
AND IN 65022(255)+("4" AND IN 6
4510=254)
16 IF LEN a$=1 THEN LET d=VAL
a$
17 LET x=x+(d=2)-(d=4): LET y=
y+(d=3)-(d=1)
18 IF SCREEN$(y,x)<>" " THEN G
O TO 2000
19 PRINT AT y,x: BRIGHT 1: INK
c1;" "
20 IF SCREEN$(y+1,x)=" " THEN
PRINT AT y+1,x: PAPER c1: INK c1
21 BEEP .01,0
22 LET a$="1" AND IN 61436(2
55)+("2" AND IN 57342=254)+("3"
AND IN 49150(255)+("4" AND IN 5
7342=253)
23 IF LEN a$=1 THEN LET c=VAL
a$
24 LET a=a+(c=2)-(c=4): LET b=
b+(c=3)-(c=1)
25 IF SCREEN$(b,a)<>" " THEN G
O TO 1000
26 PRINT AT b,a: BRIGHT 1: INK
c2;" "
27 IF SCREEN$(b+1,a)=" " THEN
PRINT AT b+1,a: PAPER c2: INK c2
28 BEEP .01,10
29 GO TO 100
3000 PRINT AT b,a: FLASH 1;" "
3010 LET s1=s1+1
3020 GO TO 3000
3030 PRINT AT y,x: FLASH 1;" "
3040 LET s2=s2+1
3050 FOR f=30 TO 60: BEEP .001,f
: NEXT f
3060 PRINT INVERSE 1;AT 9,11;"Jo
gador 1:";s1
3070 PRINT INVERSE 1;AT 11,11;"Jo
gador 2:";s2
3080 FOR f=60 TO 30 STEP -1: BEE
P .001,f: NEXT f
3090 IF s1>9 OR s2>9 THEN GO TO
4000
3100 IF INKEY$<>" " THEN GO TO 30
40
3110 PRINT AT 20,3: INVERSE 1;"C
ARRIQUE EM QUALQUER TECLA"
3120 IF INKEY$<>" " THEN GO TO 10
3130 GO TO 3060
3140 PRINT FLASH 1;AT 13,10;"Jog
ador ";(s2=10)+1;" GANHOU"
3150 INPUT "Outro jogador? ";LINE
a$
3160 IF a$="" THEN GO TO 4010
3170 IF a$(1)<>"n" THEN RUN
3180 STOP

```



```

8000 BORDER 0: PAPER 0: INK 7: C
L5
8010 PRINT "Este e a ."
8020 PRINT TAB 13;"PERSEGUIÇÃO"
8030 PRINT "O objectivo da PER
SEGUIÇÃO e empurrar o outro Jo
gador para uma parede."
8040 PRINT "Cada Jogador deika
uma parede e não pode parar at
e que um se destruido."
8050 PRINT TAB 12;"CONTROLES"
8055 PRINT TAB 14;"^ < > v"
8060 PRINT "Jogador 1, 2 0 U
R"
8070 PRINT "Jogador 2, 0 0 P
L"
8080 PRINT TAB 14;"^ < > v"
8100 INPUT "Cor do Jogador 17 (1
A 7) " ; c1
8110 IF c1<1 OR c1>7 THEN GO TO
8100
8120 INPUT "Cor do Jogador 27 (1
A 7) " ; c2
8130 IF c2<1 OR c2>7 THEN GO TO
8120

```

```

8140 RETURN
9000 FOR f=0 TO 7 STEP 2: POKE U
SR "a"+f,170: POKE USR "a"+f+1,0
5: NEXT f
9010 FOR f=0 TO 6: POKE USR "b"+
f,128: NEXT f: POKE USR "b"+7,25
5
9020 RESTORE : FOR f=0 TO 7: REA
D a: POKE USR "c"+f,a: POKE USR
"d"+7-f,a: NEXT f
9030 DATA 0,0,8,20,34,0,0,0
9040 RETURN

```

#### GRAFICOS:

Linha 50 - G + teclas 6/B

Linhas 140, 190 - G + tecla A

Linhas 8055,8080- G + teclas C,D

## ROTINAS DE APLICAÇÃO NO ZX SPECTRUM

«Acontece muitas vezes que, ao corrigir ou alterar um programa, este não tem espaço entre linhas sucessivas ou, se tem, não fica tão compreensível com as novas linhas a adicionar. Isto torna a tarefa do programador bastante demorada e cansativa.» (Hugo Assumpção/Lisboa).

### ROTINA 1 — «REORD - 9991»

HUGO ASSUMPÇÃO

Serve para reordenar linhas, com o «salto de linha» desejado: geralmente de 5 em 5 ou de 10 em 10. Tem a limitação de ter de se alterar os GO TOs e GO SUBs pontualmente, isto é, na própria linha.

```

9990 STOP
9991 LET a=PEEK 23638+256+PEEK b
9992 INPUT "1ª Linha" a
9993 INPUT "2ª Linha" b: IF b<0 OR b>=
9994 THEN GO TO 9992
9995 LET c=PEEK (x+21+256+PEEK
9996 IF PEEK a+256+PEEK (x+1)=
9997 THEN LET b=a
9998 IF PEEK a+256+PEEK (x+1)=99
9999 THEN STOP
10000 IF PEEK a+256+PEEK (x+1)=
10001 LET a=a+c: GO TO 9990
10002 LET b=b+c: GO TO 9990
10003 LET c=c+1: POKE 1+256
10004 LET c=255-1: POKE 1+256
10005 PRINT INVERSE 1: INVER
a e b: LIST a

```

#### NOTAS:

- Linha 9990 — Impede que o programa principal siga para este
- » 9991 — Calcula o endereço do início do programa.
- » 9992 — Introdução dos saltos de linha.
- » 9993 — Verifica se a linha a alterar é a primeira deste sub-programa e pára se for.
- » 9994 a 9996 — Corrige e actualiza a linha do programa.
- » 9997 — Salta para a linha seguinte a corrigir.
- » 9998 — Retorna o ciclo

### ROTINA 2 — «ERASE LINE GOTO 9991»

HUGO ASSUMPÇÃO

Permite apagar blocos de linhas de um programa. É também útil para quem trabalha com subrotinas standard, isto é, começando e acabando em determinada linha do programa.

```

9990 STOP
9991 LET a=PEEK 23638+256+PEEK b
9992 INPUT "1ª Linha" a
9993 INPUT "2ª Linha" b: IF b<0 OR b>=
9994 THEN GO TO 9992
9995 LET c=PEEK (x+21+256+PEEK
9996 IF PEEK a+256+PEEK (x+1)=
9997 THEN LET b=a
9998 IF PEEK a+256+PEEK (x+1)=99
9999 THEN STOP
10000 IF PEEK a+256+PEEK (x+1)=
10001 LET a=a+c: GO TO 9990
10002 LET b=b+c: GO TO 9990
10003 LET c=c+1: POKE 1+256
10004 LET c=255-1: POKE 1+256
10005 PRINT INVERSE 1: INVER
a e b: LIST a

```

#### NOTA:

- Linha 9991 — Calcula o endereço do início do programa principal e dá entrada das primeira (a) e última (b) linhas.
  - » 9992 — cL significa o Comprimento da Linha a examinar. Verifica se está perante a 1.ª linha a apagar(a) e, se sim, introduz o seu endereço em «a».
  - » 9993 — Verifica se a linha presente é 9990. Se sim, pára — de outra forma apagaria o subprograma.
  - » 9995 — Verifica se está perante a última linha a apagar (b) e, se sim, coloca o seu endereço em «b».
  - » 9997 — Salta para o endereço da linha seguinte e retoma o ciclo.
  - » 9999 — Calcula a diferença entre o 1.º endereço de «a» e o último de «b». Guarda-o em «c».
- Atribui à primeira linha a alterar, «a», comprimento igual a «c». Pede para apagar a linha «a».

IMPORTANTE — A primeira e a última linha do programa têm que existir sempre

Para aplicação dos rotinas 1 e 2, faça o MERGE do programa em que pretende a sua utilização.

Tenha em atenção que esse seu programa não ocupe as mesmas linhas da rotina.







```

9820 DATA 100,141,137,133,143,14
9821 DATA 100,138,128,143,143,13
9822 DATA 100,128,128,128,128,12
9823 DATA 100,128,128,133,143,13
9824 DATA 100,128,128,128,128,12
9825 DATA 100,128,137,128,137,12
9826 DATA 100,131,136,133,128,13
9827 DATA 100,141,128,128,133,12
9828 DATA 100,131,136,128,133,13
9829 DATA 100,131,136,128,133,12
9830 DATA 100,141,128,133,141,13
9831 DATA 100,131,128,128,134,12
9832 DATA 100,137,128,133,140,12
9833 DATA 100,131,136,128,137,12
9834 DATA 100,131,136,128,143,12
9835 DATA 100,131,138,129,140,13
9836 DATA 100,136,128,128,130,12
9837 DATA 100,136,128,128,130,12
9838 DATA 100,137,128,133,128,12
9839 DATA 100,128,128,129,131,13
9840 DATA 100,134,128,128,128,13
9841 DATA 100,131,136,128,137,12
9842 DATA 100,140,128,133,133,13
9843 DATA 100,131,138,133,140,13
9844 DATA 100,131,136,133,131,13
9845 DATA 100,131,136,133,128,12
9846 DATA 100,131,136,133,128,13
9847 DATA 100,131,136,133,131,13
9848 DATA 100,131,136,133,131,13
9849 DATA 100,131,136,133,132,13
9850 DATA 100,128,136,133,131,13
9851 DATA 100,136,128,128,136,12
9852 DATA 100,129,138,132,128,12
9853 DATA 100,132,138,133,136,12
9854 DATA 100,128,128,133,128,12
9855 DATA 100,132,138,136,136,12
9856 DATA 100,128,136,133,134,12
9857 DATA 100,131,136,133,128,13
9858 DATA 100,131,136,133,140,13
9859 DATA 100,131,136,133,132,13
9860 DATA 100,131,136,133,140,13

```

```

9861 DATA 100,131,136,128,134,12
9862 DATA 100,138,130,128,136,12
9863 DATA 100,128,138,133,128,13
9864 DATA 100,128,136,133,133,12
9865 DATA 100,128,138,136,136,13
9866 DATA 100,128,138,128,143,12
9867 DATA 100,128,138,129,140,13
9868 DATA 100,131,138,128,137,12
9869 DATA 100,130,128,133,128,12
9870 DATA 100,128,137,128,137,12
9871 DATA 100,128,136,128,128,13
9872 DATA 100,141,136,128,133,12
9873 DATA 100,128,128,128,128,12
9874 DATA 100,131,136,133,140,12
9875 DIM t(65,9): FOR w=1 TO 65:
  FOR u=1 TO 9: READ t(w,u): NEXT
  u: NEXT w
9880 PRINT AT 10,6:"GRAVAR ARRAY
  NUMERICO"
9885 SAVE "LARGECHARS" DATA T()
9886 PRINT AT 10,6:"VERIFICAR AR
  RAY NUMERICO"
9887 VERIFY "LARGECHARS" DATA T()

```

## LISTAGEM 2

```

9910 IF LEN Z$>10 THEN RETURN
9915 DIM w$(3,30)
9920 FOR u=1 TO LEN Z$
9925 IF CODE Z$(u)>96 AND CODE Z
  $(u)<123 THEN LET Z$(u)=CHR$(100
  OR Z$(u)-32)
9930 IF CODE Z$(u)>96 OR CODE Z$
  (u)<32 THEN RETURN
9935 LET v=0
9940 FOR w=1 TO 3: FOR u=1 TO 3
9945 LET v=v+1
9950 LET w$(w,((u-1)*3+v))=CHR$(t
  (CODE Z$(u)-31,v))
9955 NEXT u: NEXT w: NEXT u
9960 PRINT w$(1)/w$(2)/w$(3): PR
  INT
9965 RETURN

```

## PROGRAMA MAT MAT

Correções (n.º 18, pág. 11)

HUGO ASSUMPÇÃO, o autor deste programa, chama a aten-  
ção para algumas alterações:

Linha 1 será 1 PAPER 7 : BORDER 7 : INK 0 : CLS  
Eliminar as linhas 620, 625 e 626

## NOVO LIVRO

## SPECTRUM

## • SPECTRUM MACHINE CODE MADE EASY

HOLMES, Paul • Ed. Landry • 2 volumes

Preço (fixo) ..... 480\$ cada volume



# TRANSISTORES

SPECTRUM 16 e 48 K

Adapt.: ALEXANDRE SOUSA

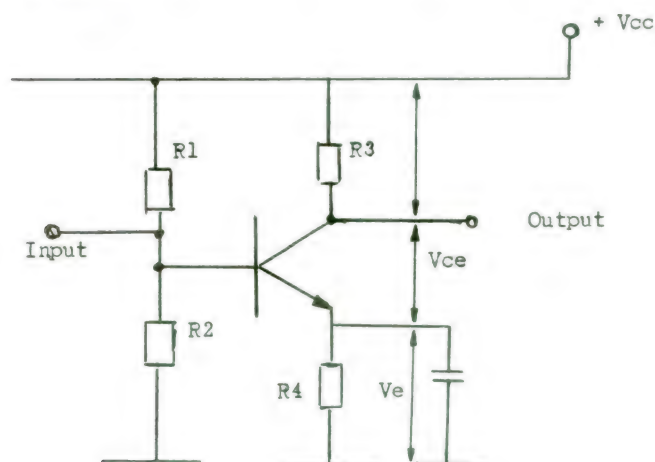
## CÁLCULO DA SELECÇÃO DO PONTO ÓPTIMO DE TRABALHO

Embora saibamos que o circuito integrado substitui, na grande parte das aplicações, os grupos de transistores usados até há pouco tempo, vamos publicar um programa de cálculo para utilizar no caso do transistor individual.

Os valores  $V_{ce}$ ,  $I_c$  e  $I_b$  podem ser encontrados nas características do componente ou em tabelas de equivalências.

O programa fornece os valores dos componentes (resistências) necessários para o bom funcionamento do circuito. Também obtém a potência de dissipação correspondente a cada resistência.

É natural que tenha de adaptar os valores obtidos aos valores standard.



```

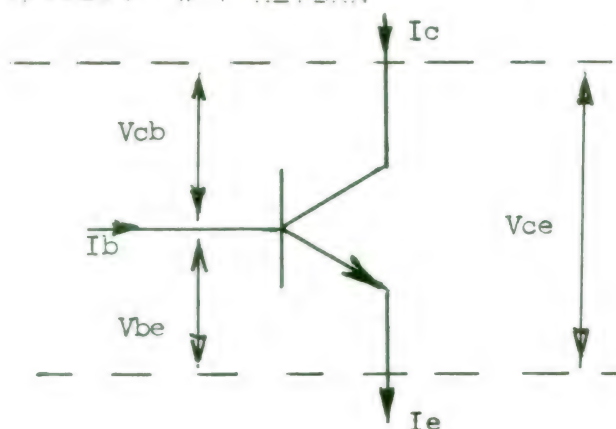
10 CLS
11 PRINT "Seleccao do ponto de operacao"
12 PRINT "de um TRANSISTOR a p artir das "
13 PRINT "caracteristicas do colector e"
14 PRINT "sendo possivel deter minar a"
15 PRINT "Tensao BASE-EMISSOR (Vbe) a par- tir dos dados do tr ansistor"
16 PRINT "Se Vbe nao e conhecido pode par- tir do valor de 0,6 V p/ trans. Silicio e 0,3 V p/ trans. Germanio"
17 PRINT "O valor tipo co p/ tensao Ve"
18 PRINT "Emissor/Gnd... 1 U"
19 PRINT "Dar entrada dos valores..."
20 INPUT "Colector/Emissor... (U)" : Vce = VAL
21 INPUT "Corr. colector... (mA)" : Ic = VAL
22 INPUT "Corr. Base... (uA)" : Ib = VAL
23 INPUT "Base/Emissor... (U)" : Vbe = VAL
24 INPUT "Emissor/(tensao)... (U)" : Ve = VAL
25 CLS : PRINT TAB 3; "Operacao do transistor" : PRINT TAB 4; "
26 PRINT "Colector/Emissor =" : TAB 20; Vce; TAB 31; "V"
27 PRINT "Corrente - Colector" : TAB 20; Ic; TAB 30; "mA"
28 PRINT "Corrente - Base =" : TAB 20; Ib; TAB 30; "uA"

```

```

29 PRINT "Base-Emissor =" : TAB 20; Vbe; TAB 31; "V"
30 PRINT "Emissor/GND =" : TAB 20; Ve; TAB 31; "U"
31 LET Ib = Ib/1000 : LET Ip = 10*Ib
32 LET Vp = Vbe + Ve : LET Vbb = 2*Vce + Ve : LET It = Ic + Ip
33 LET r1 = (Vbb - Vp)/Ip : LET r1 = INT (r1/.001+.5)
34 LET r2 = Vp/(Ip - Ib) : LET r2 = INT (r2/.001+.5)
35 LET r3 = (Vbb - Vce - Ve)/Ic : LET r3 = INT (r3/.001+.05)
36 LET r4 = Ve/(Ic + Ib) : LET r4 = INT (r4/.001+.5)
37 LET p = (Vbb - Vp)*Ip/1000 : GO SUB 300 : LET a# = p#
38 LET p = Vp*(Ip - Ib)/1000 : GO SUB 300 : LET b# = p#
39 LET p = (Vbb - Vce - Ve)/1000 : GO SUB 300 : LET c# = p#
40 LET p = Ve*(Ic + Ib)/1000 : GO SUB 300 : LET d# = p#
41 PRINT : PRINT "Resistencia (Ohm)..."
42 PRINT TAB 3;"R1";TAB 16-LEN STR# r1/r1;TAB 24;a#
43 PRINT TAB 3;"R2";TAB 16-LEN STR# r2/r2;TAB 24;b#
44 PRINT TAB 3;"R3";TAB 16-LEN STR# r3/r3;TAB 24;c#
45 PRINT TAB 3;"R4";TAB 16-LEN STR# r4/r4;TAB 24;d#
46 PRINT : PRINT "P.Alim.=";Vb b;" V / " ; It;" mA"
47 PRINT AT 21,0;"Enter para novo calc. ou STOP"
48 PAUSE 0 : IF CODE INKEY$=13 THEN CLS : GO TO 24
49 IF CODE INKEY$=226 THEN STO P
50 GO TO 260
51 REM potencia
52 IF P<=0.125 THEN LET p#="0.125 U": RETURN
53 IF P<=0.25 THEN LET p#="0.25 U": RETURN
54 IF P<=0.5 THEN LET p#="0.5 U": RETURN
55 IF P<=1 THEN LET p#="1 U": RETURN
56 IF P<=2.5 THEN LET p#="2.5 U": RETURN
57 IF P<=5 THEN LET p#="5 U": RETURN
58 IF P>5 THEN LET p#=STR# INT (P+.5)+" U": RETURN

```





## NOVOS PROGRAMAS

SPECTRUM

### JOGOS

CÓDIGO\*/PREÇOS

- **APOCALYPSE (48 K)** — Um jogo de estratégia que exige muita perícia e lógica. Um jogador terá de escolher uma época histórica e possui 3 meios de ataque/defesa: exército, armada e mísseis nucleares. O jogo pode decorrer num período pré-nuclear se todos os jogadores forem de acordo e tomarão então o papel de líderes do mundo. **IP/600\$00**
- **DOOMSDAY CASTLE (48 K)** — Neste castelo habita SCARTHAX, um ser demoníaco que esconde as suas pedras mágicas. A ajudá-lo há umas criaturas estranhas (URKS) que libertam radiações "theta" e outros seres também perigosos (GARTHROGS, ORPHCS, NEUCLOUDS, GOOGLY BIRDS). Você possui uma cápsula auto-controlada, um míssil e um campo magnético e somará pontos conforme as pedras que conseguir. **IP/400\$00**
- **EVOLUTION (48 K)** — Imagine-se na terra há 3 500 anos atrás: vulcões em erupção, céu carmesim, sol quentíssimo, falta de oxigénio... Ocorrem assim os primeiros acontecimentos da origem do mundo e da vida. Começa a evolução, cujas fases você vai seguir até ao aparecimento do homem. **IP/400\$00**
- **FIGHTER PILOT (48 K)** — Simulação de um voo baseado num F15 da Força Aérea dos E.U.A., com 8 opções: aterragem, voo de treino, prática de 1 combate aéreo, combate aéreo, aterragem c/ nevoeiro, voos c/ ventos contrários, pilotagem de perícia, controlo. Gráficos 3 D e compatibilidade c/ Joystick (Kempston). **I/400\$00**
- **PINBALL (16 K)** — Simulação das máquinas "flippers". Lance a bola e esforce-se por não a perder, de modo a atingir o máximo de pontuação. **IP/400\$00**
- **SPACE SHUTTLE (48 K)** — simulação de um voo espacial, c/ gráficos de alta resolução — indicadores c/ todas as informações necessárias. Pilote a sua nave e coloque-a na posição ideal para recolher 1 satélite à deriva. Depois aterre sem motores no deserto das Areias Brancas e, no final, ficará a saber se os seus passos foram ou não correctos e a pontuação atribuída. **I/400\$00**

### EDUCAÇÃO

- **ALGB SEC (48 K)** — Equações do 2.º grau (9.º ano unificado). Trinómio do 2.º grau (11.º ano). Números complexos: operações na forma algébrica e na forma trigonométrica. **P/600\$00**
- **BIOLOGIA (48 K)** — Contém sub-programas com: revisões de matérias pertinentes c/ cerca de 100 diagramas e textos complementares. Problemas para determinação de genotipos. Questões de escolha múltipla (peça a solução se não acertar). Natureza de substâncias. Esclarecimentos sobre diagramas. Alusões a experiências científicas. Algumas das matérias: Aparelhos circulatório, digestivo, respiratório..., reprodução, metabolismo, sistema nervoso, hereditariedade, nutrição, evolução e selecção natural das espécies, classes, fotossíntese, etc... **I/600\$00**
- **FRACÇÕES (48 K)** — Máximo divisor comum; mínimo múltiplo comum; operações c/ fracções — 2.º ano (Ciclo). **P/600\$00**
- **FUNÇÕES (48 K)** — Gráficos e funções nas formas:  $y = f(x)$ ;  $x = x(t)$  e outras — Ensino Superior. **P/600\$00**
- **GEOMETRIA DESCRITIVA (48 K)** — Toda a nomenclatura c/ apresentação gráfica de planos, intercepções, etc. Possui 1 teste sobre a situação dos pontos em relação aos planos de projecção — 10.º ano. **P/600\$00**
- **HOMOTETIA (48 K)** — Apoio ao ensino da homotetia (teoria e prática) — 8.º ano. **P/600\$00**
- **LÓGICA (48 K)** — Construção de tabelas de verdade c/ duas e três variáveis — 10.º ano. **P/600\$00**
- **FÍSICA (48 K)** — Contém sub-programas c/: revisões de matérias essenciais, c/ cerca de 250 diagramas e textos. Problemas c/ gráficos e cálculos matemáticos. Equações de escolha múltipla. Alguns temas: Leis da física, mecânica, electricidade, magnetismo, pressão, luz, electrostática, radioactividade, etc. **I/600\$00**
- **RECTA (48 K)** — As diversas formas de definir uma recta e as correspondentes equações (teoria e prática) — 10.º ano. **P/600\$00**
- **SUC,  $f(x)$  (48 K)** — Limites de vários tipos de sucessões. Estudo de funções  $f(x)$  e seus gráficos — 11.º e 12.º anos. **P/600\$00**
- **TR GEOM (48 K)** — Apoio ao ensino de translações, rotações, simetrias axiais (teoria e prática) — 7.º ano. **P/600\$00**

### UTILITÁRIOS

- **THE KEY (16/48 K)** — Faz a cópia de outros programas **I/1 000\$00**

\* Códigos: P — programa e instruções em português.

I — programa e instruções em inglês.

IP — programa em inglês e instruções em português.



# ??? MERCADO Z80 ???

Lembram-se da proposta do sócio José Gorda?

Aproveitando as suas ideias, o MERCADO Z80 será uma secção do CLUBE Z80 que **empresta aos seus sócios PROGRAMAS e LIVROS** para consulta e melhor conhecimento / aproveitamento de microcomputadores.

Depois de analisarmos pormenorizadamente as implicações desta iniciativa, excluindo a hipótese de os Sócios do Mercado (S. M.) copiarem e fotocopiarem programas e livros, optámos por estruturar o MERCADO Z80 do modo que nos parece mais funcional.

Assim, as «normas» que apresentamos em baixo parecem-nos, à partida, concretizáveis. Contudo, dado que não prevermos qual será a adesão a esta iniciativa (apenas 6 sócios

pronunciaram favoravelmente), não sabemos também se teremos as condições necessárias para responder aos interessados. Referimo-nos essencialmente ao tempo que será necessário para deslocações aos CTT's, controlo de materiais em circulação, controlo de «contas correntes», etc..

Efectivamente, se o número de aderentes for grande, sere-mos obrigados a empregar uma pessoa exclusivamente nesta tarefa, o que implicará outros problemas...

Por isso, o CLUBE Z80 faz-lhe um apelo: Depois de tomar conhecimento das «normas» do MERCADO Z80, se estiver interessado em «entrar» nele, confirme-nos por escrito até ao dia 16 de Maio. P.F. anexe um envelope selado e endereçado a si próprio — o CLUBE Z80 informá-lo-á da data de abertura do MERCADO e do n.º de aderentes. Para seu próprio interesse, comunique-nos também a sua opinião quanto ao regulamento do MERCADO Z80 que apresentamos a seguir:

1 — O MERCADO Z80 funciona exclusivamente para sócios do CLUBE Z80.

— Os produtos que o MERCADO Z80 empresta:

— Livros anunciados no Clube Z80 (há 1 exemplar de cada).

— Programas de **JOGOS** existentes no CLUBE Z80 (há 5 exemplares de cada).

3 — O MERCADO Z80 funciona apenas por correspondência (CTT), mesmo para sócios residentes no Porto. (As instalações do CLUBE e o pessoal disponível não permitem o atendimento directo).

4 — Cada sócio terá a sua «CONTA-CORRENTE» que começará com um depósito de Esc.: 1000\$00 no MERCADO Z80.

5 — Por cada encomenda enviada, o MERCADO Z80 cobrará ao sócio uma taxa de Esc.: 100\$00.

6 — Na «conta-corrente» do sócio, por cada pedido, debitar-se-ão 3 tipos de despesas:

— Embalagem Postal (20\$00 a 30\$00).

— Portes dos CTT's.

— 100\$00 (taxa de utilização dos produtos).

NOTA: Estas despesas não serão cobradas directamente pelos CTT's, pois esse processo agrava os custos. Contudo, cada sócio pode optar e informar-nos como prefere.

7 — Quando a «Conta-Corrente» deixar de ter saldo positivo, o S. M. deverá renovar o s/ depósito (1000\$00). Ele

próprio deverá controlar a sua «C.C.» pois nem sempre o CLUBE Z80 tem hipóteses de o avisar.

8 — Para além da «conta-corrente», o S.M. depositará no MERCADO Z80 a quantia de Esc. 2 000\$00 para salvaguardar o caso de não devolver o material pedido. Essa quantia será devolvida ao S.M. quando ele desistir do MERCADO Z80.

## 9 — Limites de produtos pedidos

O seu pedido nunca pode ultrapassar 5 UNIDADES (entre livros e cassettes) de cada vez. Quanto a livros, não serão emprestados mais que dois de cada vez (só há 1 exemplar de cada!).

Ex.: 2 livros + 3 cassettes

1 livro + 4 cassettes

5 cassettes

## 10 — Tempo para utilização

Os produtos recebidos poderão ficar nas mãos do sócio durante 15 dias, findos os quais deverão ser devolvidos ao MERCADO Z80 em ESTADO DE CONSERVAÇÃO e FUNCIONAMENTO idêntico àquele em que foram enviados.

**NÃO ESQUEÇA: Confirme até 16 de Maio a sua adesão ao MERCADO Z80.**

## PROGRAMA MERGE ECRAN

(n.º 17, pág. 7)

● HUGO ASSUMPÇÃO pergunta: "Não há possibilidade de transformar as linhas 100 a 180 em código-máquina? É que o transporte do ecran para a memória, em BASIC, de mora cerca de 3 minutos. Ao passo que o inverso é imediato (mas a preto e branco, visto que os atributos não são transferidos). Se sim, como? Usando a subrotina apresentada no n.º 10, pág. 6?"

● FERNANDO PRECES, o autor de MERGE ECRAN, responde: "Este programa foi escrito para simples demonstração, procurando cativar os leitores para a linguagem-máquina. A rotina em BASIC pode, de facto, ser substituída e aqui vai a respectiva alteração. Acrescente:

```
62 LET x = 32722
```

```
65 FOR n = x TO x + 12
```

```
70 READ b: POKE n,b
```

```
75 PRINT n,PEEK n
```

```
80 NEXT n
```

```
90 REM *****
```

```
100 REM Transporte do ecran para a memória, programado em cód.-máq.
```

```
105 REM (GO TO 110)
```

```
110 RANDOMIZE USR 32722
```

```
120 STOP
```

```
215 REM (GO TO 220)
```

```
310 DATA 33,0,64,17,144,101,1,0,24,237,176,201,0
```

Como pode verificar, a rotina em C.M. é muito semelhante à da pág. 6 do boletim 10, c/ pequenas diferenças pelo facto de ter sido escrita para o ZX81."







# CLUBE Z<sub>80</sub>

## INSCRIÇÃO COMO ASSOCIADO

O **CLUBE Z<sub>80</sub>** está aberto a todos os utilizadores de microcomputadores.

A intenção de associar os entusiastas das micro-máquinas, é exclusivamente a de permitir:

- 1 — PUBLICAÇÃO DE UM JORNAL MENSAL, onde sejam publicados programas de uso geral ou específico como no caso da educação.
- 2 — PROMOVER TROCAS DE PROGRAMAS, e trocas de experiências; tanto no caso do **Software** (programação), como no caso do **Hardware** (electrónica).
- 3 — PROMOVER DESCONTOS NA AQUISIÇÃO DE PROGRAMAS.
- 4 — LANÇAR CURSOS DE PROGRAMAÇÃO EM BASIC — PASCAL OU OUTRAS LINGUAGENS E DIVULGAR O USO DE LINGUAGEM MÁQUINA.

NOME .....

IDADE ..... COMPUTADOR TIPO .....

PROFISSÃO .....

ENDEREÇO .....

TELEF. ....

ASSINATURA ANUAL — Esc. 1 500\$00 ☐

ASSINATURA SEMESTRAL — Esc. 750\$00 ☐

CHEQUE OU VALE DO CORREIO

N.º .....

BANCO .....

DATA ...../...../.....

JÁ SÓCIO ☐

NOVO SÓCIO ☐ → A partir do mês de ..... (inclusive)